C 20 – 1684 – 2F

IBM Systems Reference Library

Introduction to IBM Data Processing Systems

Johdanto Tietojenkäsittelyjärjestelmiin

Tämä julkaisu on käännös ohjekirjasta Introduction to IBM Data Processing Systems Form C20-1684-2, jonka on julkaissut International Business Machines Corporation, U.S.A.,

Kääntäjä: Jukka Nurmi
10 12 69 JN

Ladonta: IBM MT Composer

ESIPUHE

Kaikilla IBM:n tietojenkäsittelyjärjestelmillä on tiettyjä koosta, tyypistä, tai käyttötarkoituksesta riippumattomia yhteisiä peruskäsitteitä ja toimintaperiaatteita. Tässä julkaisussa pyritään selvittelemään näitä käsitteitä ja toimintaperiaatteita ja siten antamaan lukijalle tietokonealan perustietous. Julkaisu on tarkoitettu käytettäväksi ATK-alan peruskoulutuksessa, joka on edellytyksenä johonkin tiettyyn IBM-järjestelmään kohdistuvalle opiskelulle.

Kukin jakso kuvaa toisiinsa liittyvien käsitteiden ja toimintaperiaatteiden loogista kokonaisuutta. Yleiskuvan saamiseksi tietokoneista voi lukija tutustua järjestelmällisesti koko julkaisuun tai käyttää erillisiä jaksoja lähdeaineistona. Aiheita on pyritty käsittelemään yleisesti ja viittaamaan varsinaisiin laitteisiin ja järjestelmiin mahdollisimman vähän. Yksityisiin järjestelmiin viittaamalla on pyritty selventämään ja havainnollistamaan jotakin yleistä periaatetta tai käsitettä eikä vertailemaan järjestelmiä toisiinsa.

Tässä julkaisussa lukija kohtaa monia IBM tietojenkäsittelyjärjestelmiin liittyviä käsitteitä, laitteita, jne. Edelleen julkaisun eri jaksoissa käsitellään ohjelmien muistitilavaatimuksia, syöttö- ja tulostuslaitteiden ja muistilaitteiden kapasiteetteja ja nopeuksia, järjestelmiin kuuluvia erikoislaitteita jne. Tietojenkäsittelyala on kuitenkin luonteeltaan dynaamista, se muuttuu ja kehittyy hyvin nopeasti. Sen vuoksi onkin tässä yhteydessä syytä mainita IBM Systems Reference Library -käsikirjat, joista viimeisimmät alan tiedot ovat saatavissa.

SISÄLLYSLUETTELO

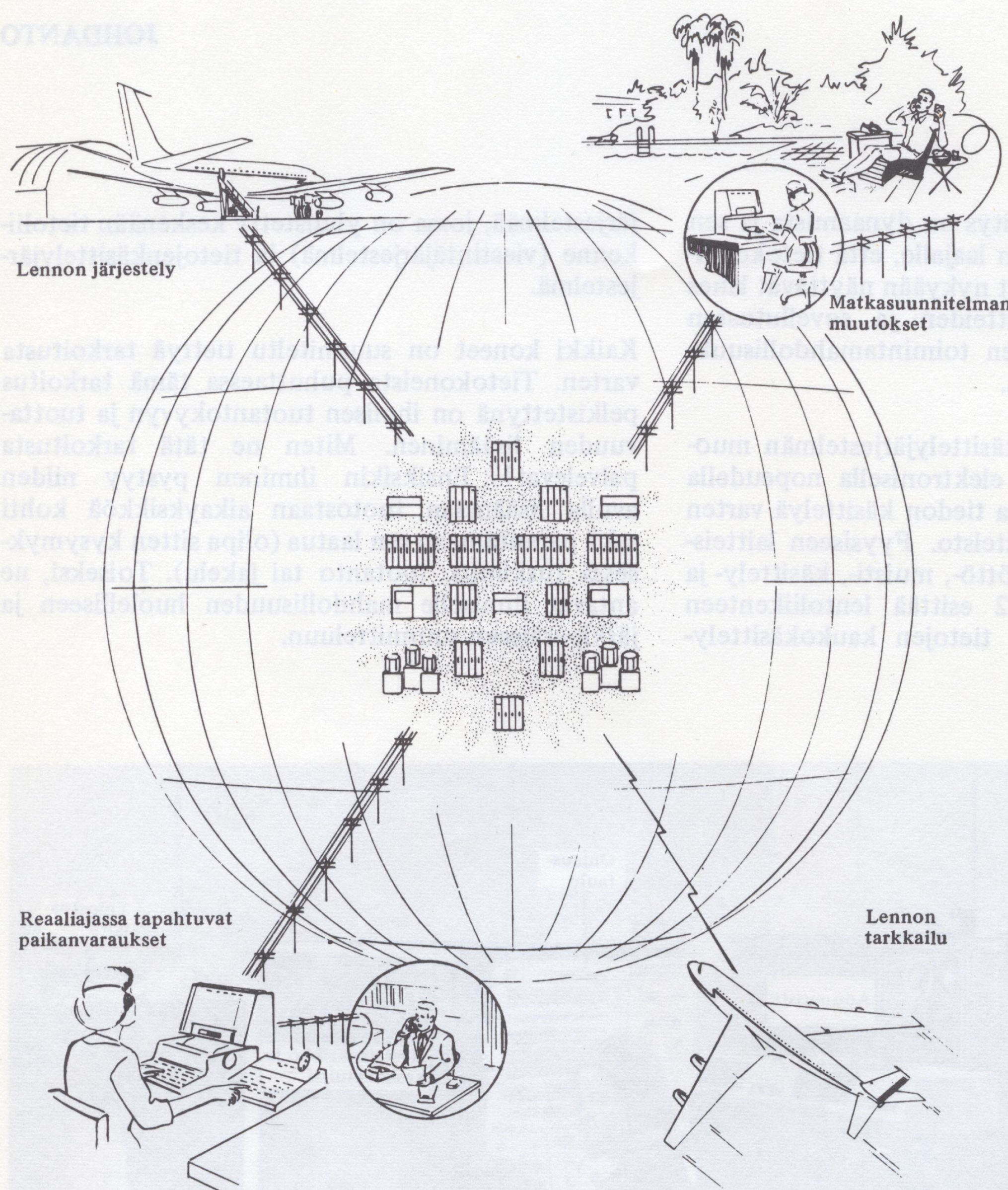
Tietojenkäsittelyalan kehitys on dynaamista ja sen vaikutukset ulottuvat niin laajalle, että tietokoneiden käyttömahdollisuudet nykyään näyttävät lähes rajattomilta. Uusien laitteiden ja sovellutusten kehitys merkitsee ihmisen toimintamahdollisuuksien jatkuvaa laajenemista.

Yleisesti ottaen tietojenkäsittelyjärjestelmän muodostavat ohjelmat sekä elektronisella nopeudella tapahtuvaa, itsetarkistavaa tiedon käsittelyä varten suunniteltu fyysinen laitteisto. Fyysiseen laitteistoon (kuva 1) kuuluu syöttö-, muisti-, käsittely- ja tulostuslaitteita. Kuva 2 esittää lentoliikenteen paikanvaraussovellutusta, tietojen kaukokäsittelyjärjestelmää, jossa on yhdistetty keskenään tietoliikenne (viestintäjärjestelmä) ja tietojenkäsittelyjärjestelmä.

Kaikki koneet on suunniteltu tiettyä tarkoitusta varten. Tietokoneista puhuttaessa tämä tarkoitus pelkistettynä on ihmisen tuotantokyvyn ja tuottavuuden lisääminen. Miten ne tätä tarkoitusta palvelevat? Ensiksikin ihminen pystyy niiden avulla lisäämään tuotostaan aikayksikköä kohti sekä parantamaan sen laatua (olipa sitten kysymyksessä tutkimus, tuotanto tai jakelu). Toiseksi, ne antavat ihmiselle mahdollisuuden huolelliseen ja järkiperäiseen suunnitteluun.



Kuva 1. IBM Systeemin/360 mallin 40 tietojenkäsittelyjärjestelmä



Kuva 2. Tietojenkäsittelysovellutus

Tietokoneet syntyivät ensisijaisesti tyydyttämään jatkuvasti kasvavaa informaatiotarvetta yhä mutkikkaammiksi kehittyvissä olosuhteissa. Teollisuustalouden kehittyessä 19. vuosisadalla, kävi selväksi, että kasvaneet markkinat vaativat massatuotantotekniikkaa. Teollisuus alkoi koneellistua. Kävi mahdolliseksi tuottaa yhä enemmän hyödykkeitä yhä pienemmällä ihmistyömäärällä.

Viimeksi kuluneen neljännesvuosisadan aikana kehityksen vauhti on edelleen kiihtynyt. Tiede on siirtynyt etualalle. Tutkimustoiminta nielaisee vuosittain miljardeja dollareita. Tekninen kehitys on kiihdyttänyt yritysten kasvua. Palveluelinkeinoissa kasvu on ollut moninkertaista. Kuluttajan koko ajattelutapa ja kulutuksen rakenne ovat kokeneet perusteellisia muutoksia.

Muutosten vähitellen voimistuessa niiden vaikutus alkoi näkyä monin tavoin. Tiedot saivat aivan uuden merkityksen. Informaation tarve lisääntyi voimakkaasti. Tämä puolestaan merkitsi erilaisten toimistotehtävien moninkertaistumista. Syntyi eräänlainen epätasapaino, sillä toimistotyön meka-

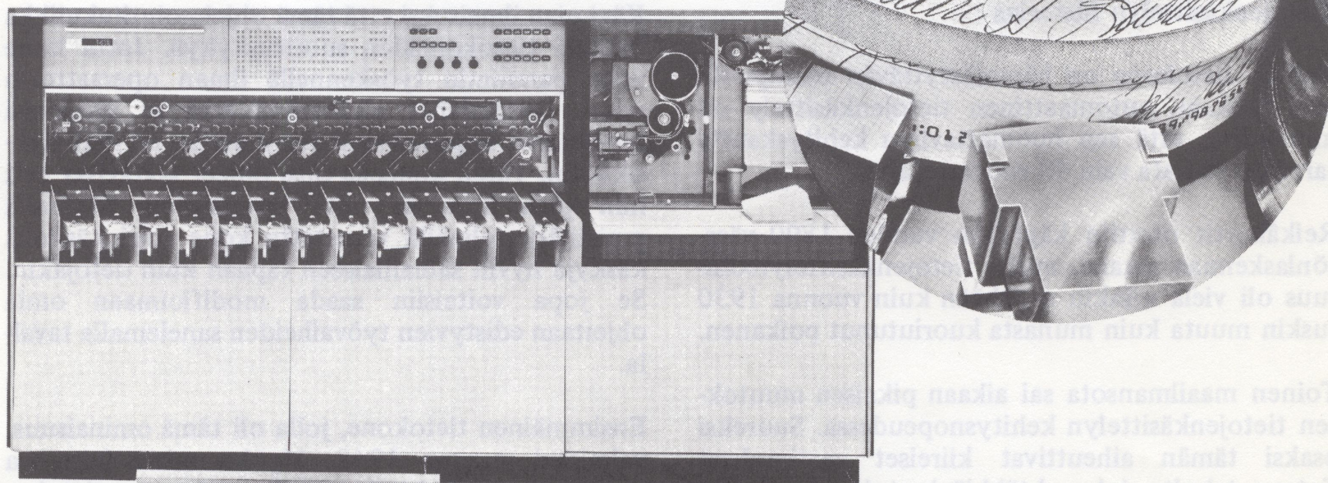
nisaatio oli jäänyt jälkeen tuotantopuolella tapahtuneesta kehityksestä ja näytti siltä, että kaikenlaisten asiapapereiden käsittely hukuttaisi alleen kaiken tuottavan toiminnan.

Tulevaisuus tarjoaa valtavia kehitysmahdollisuuksia. Eräs esimerkki siitä, mitä liike-elämän piirissä on jo tehty, on magneettisen musteen kehittäminen pankkeja varten. Yhdysvalloissa on vuosittain liikkeellä arviolta 10 miljardia shekkiä. Tämä merkitsee pankeille valtavaa tietojenkäsittelytehtävää, koska jokainen lunastettu shekki on käsiteltävä vähintään kuusi kertaa ennen kuin se voidaan mitätöidä. Konttorikoneiden käyttöönottokaan ei ratkaissut tätä ongelmaa kokonaisuutena, tarvittiin edelleen operaattoreita tietojen muuntamiseksi koneiden vaatimaan muotoon.

Tietokoneiden valmistajien ja Yhdysvaltain pankkiyhdistyksen (ABA, American Bankers Association) yhteistyössä kehittämän magneettimerkkien tunnistusmenetelmän avulla kone voi, kuten ihminenkin, lukea tiedot suoraan shekkilomakkeelta (kuva 3). Tietokoneiden valmistajien, lomakepainojen ja ABA:n keskinäisen sopimuksen mukaan pankkitositteet (shekit, talletuskuitit jne.) voidaan painaa magneettimusteella. Esipainettu tieto, jonka kone siis voi lukea suoraan lomakkeelta, sisältää pankin tunnuksen, tallettajan tilinumeron jne. Shekin tai talletuskuitin tullessa käsittelyyn siihen merkitään ko. määrä erityisillä merkintälaitteilla ja tämän jälkeen lomake on valmis kulkemaan koko prosessin läpi.

Toimistotehtävien ja johtamismenetelmien kasvavan automatisointitarpeen lisäksi tietojenkäsittelytarve kasvaa jatkuvasti teknisen kehityksen ja tieteellisen tutkimuksen kiihtyvässä vauhdissa. Informaatiotarpeet ovat suunnattomat. Yhä kasvavassa määrin turvaudutaan tietokoneisiin yritystoiminnassa, julkisessa hallinnossa, tutkimustyössä ja suunnittelussa. Näiden tarpeiden tyydyttämiseksi ovat tietokonekeskukset alkaneet lisääntyvässä määrin tarjota tietokoneen osituskäyttömahdollisuutta (time-sharing). Käyttäjät voivat antaa tietokoneelle ongelmia ratkaistaviksi, pyytää informaatiota, syöttää tietoja käsiteltäväksi etäispäätteiden avulla, jotka voivat sijaita vaikkapa tuhansien kilometrien päässä itse koneesta. Saman yhtiön puitteissa toimivista osituskäyttöjärjestelmistä esimerkkejä ovat lento- ja motelliyhtiöiden paikanvaraussovellutukset.

Kaksi muuta aluetta, joilla kehitys on ollut huomattavaa ovat kuvan ja äänen sisällyttäminen tietojenkäsittelyjärjestelmään. Kuvan käsittelystä



Kuva 3. Magneettisten merkkien tunnistus - IBM 1419 magneettimerkkien lukija

(image processing) esimerkkinä on Mariner IV:n Marsista lähettämien (valo)kuvien käsittely. Mikrofilmin tutkainlaitteistoon yhdistettiin automaattinen muuntojärjestelmä, joka tulkitsi kuvien tummat ja vaaleat kohdat ykkösinä ja nollina tietokoneen muistiin tallentamista varten. Lopullisten kuvien näyttö tapahtui erityisten katselulaitteiden avulla.

Vaikka nykyään on mahdollista olla yhteydessä tietokoneen kanssa puhutun sanan välityksellä, tietokoneen toiminta ihmisen sanelun mukaan on vielä kokeiluasteella. Sen sijaan päinvastainen kommunikaatio, tietokoneen 'puhuminen' ihmiselle, toimii nyt jo esim. New Yorkin pörssissä, jossa tietokone ilmoittaa pyydettäessä päivän viimeisimmät noteeraukset 'tallennetun' äänen välityksellä. Tällainen 'pörssitiedotus' kootaan tietokoneen muistiyksiköihin etukäteen tallennetuista puhutuista sanoista. Menetelmä on hieman samankaltainen kuin nauhurissa - ero on lähinnä siinä että tietokoneeseen tallennetaan sanasto, jonka jokainen sana on käsiteltävissä erikseen kun taas nauhurin nauhalle tallennetaan tietty valmiiksi muotoiltu teksti, jonka yksityiset osat, sanat, ovat kiinteässä järjestyksessä.

Kuten alussa todettiin, tietokoneiden käyttömahdollisuudet näyttävät lähes rajattomilta - niitä ei rajoita käsillä oleva tuote tai ongelma, ei yrityksen tai laitoksen luonne. Ihminen käyttää arvostelukykyään ja tekee lopulliset päätökset, mutta perustaksi tarvitaan informaatiota ja tässä tietokoneen apu saattaa olla korvaamaton.

Tietojenkäsittelyn historiaa

Vaikka tietokone on hämmästyttävän monipuolinen työkalu, automaattinen tietojenkäsittely on niin uutta, että sen huomattavinta kehityskautta tarvitsee seurata vain 40 vuoden ajalta.

Reikäkortit otettiin käyttöön vuoden 1890 väestönlaskennan aikana, mutta tietojenkäsittelyteollisuus oli vielä niinkin myöhään kuin vuonna 1930 tuskin muuta kuin munasta kuoriutunut poikanen.

Toinen maailmansota sai aikaan pikaisen muutoksen tietojenkäsittelyn kehitysnopeudessa. Suureksi osaksi tämän aiheuttivat kiireiset vaatimukset tieteen taholta, joka yhtäkkiä joutui työskentelemään ennen näkemättömässä mittakaavassa uusien aseiden kehittämiseksi. Lentokoneiden suunnittelussa ja tykistön kehityksessä tarvittiin uusia ja suunnattoman suuria tietomääriä. Ja atomipommia valmistettaessa tiedemiehet havaitsivat olevansa vastatusten aivan uutta suuruusluokkaa olevien laskelmien kanssa.

Sekä Yhdysvalloissa että sen ulkopuolella ensimmäiset kaksi tietokonetta kehitettiin yliopistojen laboratorioissa. Ensimmäinen, ENIAC, rakennettiin Pensylvanian yliopistossa. Euroopan ensimmäinen, EDSAC, valmistui Cambridgen yliopiston laboratorioissa Englannissa.

Näissä koneissa tyhjiöputket hoitivat kytkentä- ja valvontatoimintoja, jotka aikaisemmin oli hoidettu releillä. Täten sähkömekaanisissa laskijoissa olleiden kytkimien suhteellisen hitaat liikkeet korvattiin elektronien nopealla liikkeellä. Tämän muutoksen ansiosta tuli mahdolliseksi lisätä toimintanopeutta ja suorittaa laskut 1000 kertaa nopeammin kuin aikaisemmin.

Melkein samanaikaisesti elektroniikan käyttöönoton kanssa tuli toinen suuri edistysaskel, joka lisäsi tietokoneiden toimintakykyä ja laajensi niiden sovellutusmahdollisuuksia. Tämä on niin sanotun tallennetun ohjelman käyttö.

Aluksi konekäskyt ohjelmoitiin säädettävillä ohjaustauluilla tai vaihdettavilla korteilla tai paperinauhoilla. Yksityiskohtaiset ohjeet täytyi syöttää koneeseen sitä mukaa kuin suoritettava työ edistyi. Tietokoneeseen syötetyt tiedot käsiteltiin näiden ennalta asetettujen yksiköiden sisältämien ohjeiden mukaan. Tietokone saattoi poiketa kiinteästä ohjelman järjestyksestä vain rajoitetulla tavalla.

Kävi pian ilmeiseksi että tämä ohjelmointitekniikka haittasi tietokoneiden suorituskykyä. Jotta kone voisi paremmin työskennellä ilman operaattorin apua, tiedemiehet ehdottivat että se varastoisi ohjelmansa nopeaan sisäiseen muistiin tai muistiyksikköön. Täten koneella olisi ohjeet saatavissa heti niin pian kuin se niitä tarvitsisikin. Sisäisellä muistijärjestelmällä varustettu kone voisi käsitellä käskyjä hyvin samanlaiseen tapaan kuin tietojakin. Se jopa voitaisiin saada modifioimaan omia ohjeitaan edistyvien työvaiheiden sanelemalla tavalla.

Ensimmäinen tietokone, jolla oli tämä ominaisuus, valmistui vuonna 1948. Myöhemmissä laitteissa periaatetta on kehitelty edelleen, kunnes tietokoneelle kävi mahdolliseksi muodostaa itse huomattava osa omia käskyjään. Koska tietokone kykenee

tekemään yksinkertaisia päätöksiä ja koska se kykenee modifioimaan käskyjä, käyttäjä vapautuu suuresta määrästä kallista ja toistuvaa ohjelmointia.

Samaan aikaan kun syntyi nykyisille tietokoneille niin oleellinen käsite, ohjelma, syntyi myös tietojen kaukokäsittely (teleprocessing), vaikka siitä ei juuri tätä nimitystä käytettykään vielä silloin, vaan vasta yli 10 vuotta myöhemmin. Itse asiassa tietojen kaukokäsittely pitäisi tässä tapauksessa korvata termillä tietojen kaukosiirto. Vuonna 1940 nimittäin Yhdysvaltain ilmavoimat ilmoittivat tarvitsevansa laitteen, joka automaattisesti lävistäisi reikäkortteja lennätinlinjoja pitkin siirrettyjen, reikänauhalle vastaanotettujen tietojen perusteella. Tätä tarkoitusta varten IBM kehitti reikänauhan avulla ohjattavan kortinlävistyskoneen ja reikäkortin avulla ohjattavan nauhanlävistyskoneen. Kahden viimeisen sotavuoden aikana tämä lennätinlinjoja käyttävä tietojen siirtoliikenne käsitti 4-5 miljoonaa reikäkorttia kuukausittain.

Seuraava suuri edistysaskel tietojen siirrossa oli IBM Data Transceiver (1954), laite, joka mahdollisti suoran reikäkortilta reikäkortille tapahtuvan tietojen siirron puhelinlinjoja pitkin. Linjoina voitiin käyttää myös lennätinlinjoja sekä mikro- ja lyhytaaltoradiolaitteita.

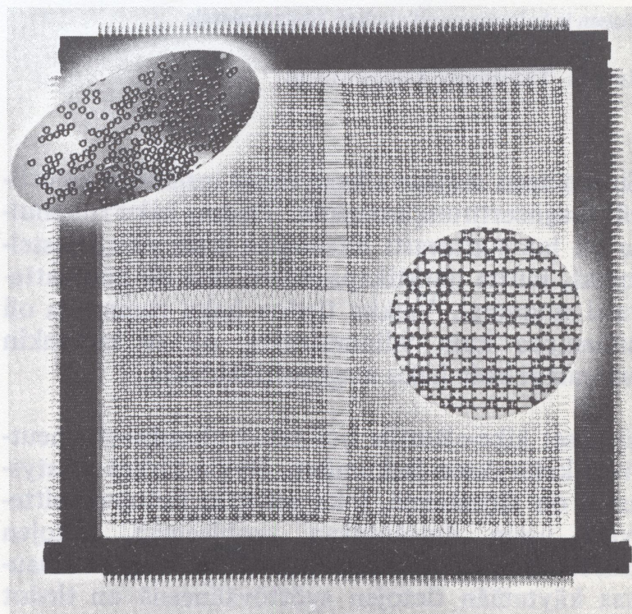
1950-luvun alkupuolella kehittyivät keskisuuret ja suuret tietojenkäsittelyjärjestelmät, jotka oli erityisesti suunniteltu ottamaan suoritettavakseen rasittavat konttorityöt, jotka ahdistivat niin monia kasvavia yhtiöitä. Vaikka nämä uudet liikekoneet ovat tietojenkäsittely.tavoiltaan oleellisesti samanlaisia kuin aikaisemmat tietokoneet, ne ovat kuitenkin erilaisia monissa ulkoisissa seikoissa. Tieteellisessä tutkimuksessa useimmat ongelmat vaativat suhteellisen vähän käsiteltäviä numerotietoja koneen intensiiviseen toimintaan. Liike-elämän tehtävissä on asianlaita useimmiten päinvastoin. Niissä tarvitaan koneita, jotka soveltuvat suunnattomaan numeroiden lukumäärään, kun taas tietojen käsittely vertailemalla on tavallisesti melko yksinkertaista.

Näissä uusissa kaupallisissa systeemeissä muutokset koskivat lähinnä kahta ongelmaa - syöttöä ja tulostusta.

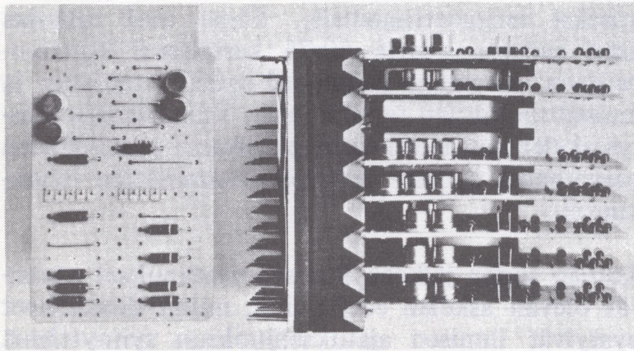
Vanhemmissa tietokoneissa oli käytetty yksinomaan reikäkortteja ja reikänauhaa tietojen syöttöön. Nyt kehitettiin menetelmä tietojen tallentamiseksi magneettinauhalle. Tämä uusi ratkaisu lisäsi syöttönopeutta 50-75 kertaiseksi korttinopeuteen verrattuna ja paransi syöttö-, tulostus- ja muistitoimintojen tehokkuutta. Viimeaikaiset edistysaskelet magneettinauhatekniikassa ovat edelleen parantaneet syöttö- ja tulostustoimintojen tehokkuutta.

Korean sodan jälkeen ihmisen vaatimukset näyttivät olevan askelen edellä siitä, mihin tietokoneet pystyivät ihmisen ajatuksenjuoksun synnyttämiä suuritöisiä loogisia ja aritmeettisia tehtäviä suorittaessaan. Tarve kasvoi erityisesti sellaisilla aloilla kuin ydinfysiikassa ja avaruustekniikassa, joissa vetypommin ja ballististen ohjusten parissa tehdyn työn probleemat saattoivat silloisten koneitten kapasiteetin kovalle koetukselle. Kävi ilmeiseksi että tietokoneissa tarvitiin yhä suurempaa nopeutta.

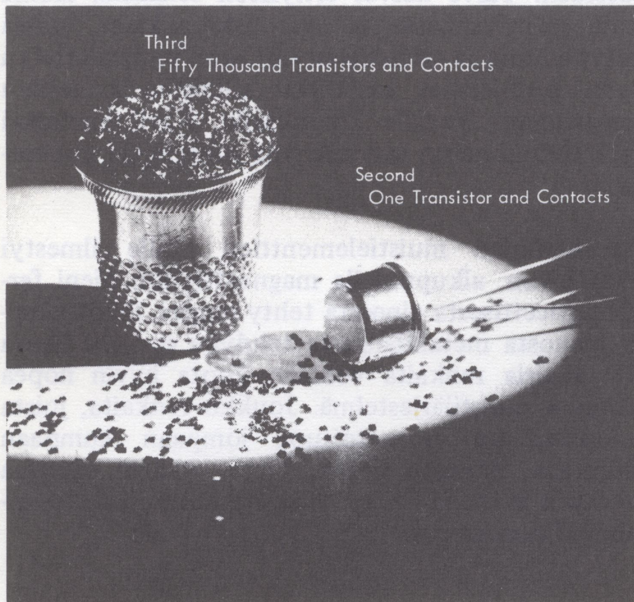
Aikaisempien muistielementtien tilalle ilmestyi 1950-luvun alkupuolella magneettiydin, pieni ferromagneettisesta aineesta tehty rengas. Pujottamalla hienosta metallilangasta tehdylle verkolle (kuva 4) tällaisia renkaita voidaan koota hyvin nopea sisäinen muistijärjestelmä. Joukko renkaita, joista jokainen on magnetoituna jompaan kumpaan suuntaan, ilmaisee tietoa. Tieto voidaan sijoittaa muistiin ja saada käsiteltäväksi muutamassa sekunnin miljoonasosassa.



Kuva 4. Magneettiydintaso



Toisen sukupolven komponentteja



Toisen ja kolmannen sukupolven komponentteja

Kuva 5. Tietokoneen komponentteja

Suunnilleen samaan aikaan toiset insinöörit kehittivät magneettirumpumuistin. Tiedon haku rummulta oli huomattavasti hitaampaa kuin ydinjärjestelmässä, mutta muistikapasiteetti taas oli huomattavasti suurempi. Vaikka tiedon haku rummulta oli hitaampaa kuin ydinmuistista, oli se kuitenkin nopeampi kuin haku magneettinauhalta.

Erilaiset liike-elämälle ominaiset olosuhteet aiheuttivat kehityksen jatkumisen. Hyvin tärkeä edistysaskel on järjestelmä, joka selvittää tietojenkäsittelyssä usein ilmaantuvan probleeman, tietojen jaksottaisen tulon. Esimerkiksi kun magneettinauhaa käytetään tietojen syöttövälineenä on tiedot koottava pitemmiksi jaksoiksi ennen niiden sisäänlukua, muussa tapauksessa kone tulee kalliiksi ja aikaa kuluttavaksi. Kun tarkastellaan rajoitusta liike-elämässä, voidaan havaita, että jokainen tietoalkio voi olla vain niin 'tuore' kuin se tietoerä (batch) johon ko. tieto on syöttöä varten liitetty. Normaalitoiminnassa saattaa näiden erien sisäänluvun välillä kulua tunteja ja joskus jopa päiviä.

Tilanne korjautui 1950-luvun puolivälissä, jolloin markkinoille ilmestyivät ensimmäiset poimintamuistilaitteet, levymuistiyksiköt, joissa voidaan päästä suoraan käsittelemään mitä tahansa tietuetta ja näin säästää aikaa normaaliin peräkkäiskäsittelyyn verrattuna. Ensimmäisten poimintamuistiyksiköiden varsinainen 'muisti' koostui magnetoiduista pyörivistä levyistä, jotka jakautuivat useampiin tietouriin. Tietoa voidaan tallentaa urille tai lukea urilta, riippumatta siitä missä järjestyksessä ne on uralle alun perin tallennettu.

Sillä välin kehitys pulssielektroniikassa ja kiinteän aineen fysiikassa tuotti uusia ja yhä parempia komponentteja.

Joissakin kytkevissä toiminnoissa tyhjiöputki korvattiin pienemmällä puolijohdediodilla, jonka etuna on vähäinen energian kulutus. Edistymistä jatkoivat pienet transistorit, joilla korvattiin tyhjiöputket. Paitsi että näitä transistoreita voidaan koota pienempikokoisiksi elementeiksi, ne ovat myös luotettavampia (kuva 5). Transistoreiden käyttöönottamisella saatiin aikaan koneita, joista usein käytetään nimitystä 'tietokoneiden toinen sukupolvi'.

Rajoitus tulee pahimmin esiin, kun haetaan nauhalta tiettyä osatietoa. Kone saattaa joutua käymään läpi koko pitkän kelan sitä etsiessään. Haku on hidasta, aikaa kuluu hukkaan.

Tiheään esiintyvät tietojen jaksottaisen sisäänluvun ja tiedon etsimisen vaatimukset aiheuttavat vakavan haitan tieteellisessäkin työssä. Kaupan piirissä vaikeus tulee paljon suuremmaksi, erityisesti kirjanpidossa.

Seuraavan teknisen kehityskauden aikana pienennettiin ja paranneltiin toisen sukupolven koneiden komponentteja. Tämä johti jo mikropiiritekniikan käyttämiseen. Tietokoneiden 'kolmas sukupolvi' käyttää mikropiiritekniikan avulla aikaansaatuja komponentteja (kuva 5).

Tiedemiehet ovat suunnitelmissaan edenneet vielä pitemmälle. Jotkut tutkivat mikroaaltoilmiöiden käyttöä laskulogiikan suoritusvälineenä. Toiset

tutkivat aineen ja elektronien käyttäytymistä äärimmäisen alhaisissa lämpötiloissa (cryogeniikka).

Kuten on aina ollut, on nytkin tarkoitus kehittää parempi, monipuolisempi ja käytännöllisempi tietokone, joka työskentelee nopeammin, varastoi enemmän tietoa, tarvitsee vähemmän virtaa, vie vähemmän tilaa ja maksaa vähemmän kuin aikaisemmin kehitetyt koneet.

Tulevaisuus

Tulevaisuuden tietokoneet tulevat väistämättömästi aiheuttamaan muutoksia lähes kaikilla inhimillisen toiminnan aloilla. Ne tulevat muuttamaan työskentelytapojamme, lisäämään oppimismahdollisuuksiamme, vaikuttamaan maanpuolustuskysymyksiin jne.

Uuden, automaattista ohjelmointia tutkivan tieteenalan tehtävänä on saada ohjelmointi helpommaksi ja joustavammkasi. Tavoitteena on rakentaa sellaisia tietokoneita ja järjestelmiä, joille voidaan antaa käskyjä normaalista, ihmisten puhumasta kielestä mahdollisimman vähän poikkeavalla kielellä. Lopullisena tavoitteena voidaan pitää koneita, jotka pystyvät lukemaan aivan tavallista painettua tekstiä sekä 'ymmärtämään' puhuttuja sanoja.

Merkittävä edistys syöttö/tulostustekniikassa on erilaisten graafisten näyttölaitteiden kehitys. Näyttölaitteet muistuttavat ulkonaisesti televisiovastaanotinta, mutta näyttölaitteen kuva on eräs tietokoneen tulostusmuoto, jonka elementit, kirjainmerkit, graafiset käyrät tms. luodaan tietokoneessa olevan ohjelman ja annettujen tietojen avulla. Joissakin tapauksissa voidaan näyttölaitteen kuvaputkella olevaa kuvaa muuttaa ns. valokynän avulla. Paitsi kuvaa, voidaan valokynän tai näyttölaitteessa olevan näppäimistön avulla muuttaa itse tietoa, (joka siis varsinaisesti sijaitsee tietokoneen muistissa).

Tulevaisuuden tietokoneet samoin kuin niiden ohjelmisto poikkeavat todennäköisesti varsin paljon nykyisistä. Keskus- ja muistiyksiköiden fyysinen koko pienenee huomattavasti, mutta nopeus ja kapasiteetti sen sijaan lisääntyvät. Nykyäänkin on jo monia järjestelmiä, joissa yksi tietokone hoitaa useita, etäällä toisistaan olevia kyselyasemia ja terminaaleja. Tällaisten laajoja verkostoja käsittävien systeemien avulla on jo pystytty integroimaan esim. suuryritysten yli koko maan ulottuva toiminta. Tämän kaltaiset fyysiset muutokset ovat luoneet uusia tarpeita. Tarvitaan koko systeemiä valvovia ohjausohjelmia, uudenlaisia käyttöohjelmia, ohjelmien prioriteettijärjestelmiä, koska kaikki toiminnot eivät ole samanarvoisia jne.

Edellä kuvatun kaltaisten verkostojen avulla voidaan kommunikaatio ja informaation käsittely hoitaa luotettavammin ja halvemmin. Tulevaisuudessa tämä saattaa merkitä esim. sitä, että maksuvälineenä käytetyt shekit häviävät - tilalle tulee tietokoneen pankeissa suorittamat siirrot tililtä toiselle. Samoin tietysti voidaan ajatella meneteltävän käteisen rahan suhteen. Mahdollisuuksia on rajattomasti.

TIETOJENKÄSITTELYJÄRJESTELMÄ

Tietojenkäsittely on annettuihin tietoihin kohdistuva sarja ennalta suunniteltuja tehtäviä ja toimintoja halutun tuloksen aikaansaamiseksi. Tähän tarkoitukseen käytetyt proseduurit ja laitteet muodostavat tietojenkäsittelyjärjestelmän (kuva 6). Laitteet voivat vaihdella: kaikki tehtävät voidaan suorittaa koneella taikka 'laitteina' saattavat olla vain kynä ja paperi. Proseduurit säilyvät silti pohjimmiltaan muuttumattomina.

IBM:llä on monia tietojenkäsittelyjärjestelmiä. Ne vaihtelevat kokonsa, rakenteensa, nopeutensa, hintansa ja käyttönsä puolesta. Mutta riippumatta käsiteltävistä tiedoista tai käytetystä laitteesta sisältyy kaikkeen tietojenkäsittelyyn ainakin kolme perusosaa:

- lähtötiedot eli syöttömateriaali syötetään järjestelmään.
- järjestelmällinen ja suunniteltu käsittely järjestelmän sisässä.
- lopputulos eli tulostusmateriaali otetaan ulos järjestelmästä.

Syöttömateriaali voi koostua minkälaisista tiedoista tahansa: kaupallisista, tieteellisistä, tilastollisista, teknillisistä jne. (kuva 7).

Käsittely suoritetaan ennalta määrätyssä perättäisessä käskyjärjestyksessä, jota kone automaattisesti

Kirjanpitojärjestelmä - laskenta kynän ja paperin avulla

Laskenta kirjanpitokoneilla

Laskenta reikäkorttikoneilla

Tietojenkäsittelyjärjestelmä

Kuva 6. Tietojenkäsittelyjärjestelmiä

Kuva 7. Tiedon lähteitä

Kuva 8. Tietojen käsittely tietokoneella

seuraa (kuva 8). Käsittelyn suunnittelu on aina inhimillistä alkuperää. Laskemisen, lajittelun, analyysin tai muiden toimintojen avulla kone saa aikaan tuloksen, jota voidaan käyttää jatkokäsittelyyn tai joka voidaan tallentaa raporttina tai tiedostona.

Ohjelmat

Jokainen tietojenkäsittelyjärjestelmä on suunniteltu suorittamaan vain määrätyn määrän ja määrätynlaisia toimintoja. Se on ohjattu suorittamaan toiminnat käskyjen mukaan. Käsky määrittää suoritettavan perustoiminnan ja tiedot, laitteet tai mekanismin, jota tarvitaan toiminnan suorittamiseksi. Kokonaista annetun työsuorituksen loppuun saattamiseksi tarvittavaa käskysarjaa sanotaan ohjelmaksi. Esimerkiksi voi tietokoneen virtapiireihin olla konstruoituna kertolasku melkein samaan tapaan kuin yksinkertaiseen pöytälaskukoneeseen on konstruoitu yhteenlasku. Täytyy olla jokin keino ohjata tietokone suorittamaan kertolasku juuri kuten yhteenlaskukonetta ohjataan nappeja painamalla. Täytyy siis olla keino antaa koneelle ohjeet, mistä se voi löytää kertolaskun tekijät.

Edelleen sisältää suhteellisen yksinkertainen kertolaskutoiminta muita toimintoja, joiden täytyy edeltää ja seurata laskutoimitusta. Kerrottava ja kertoja täytyy lukea muistiin syöttölaitteella. Tällä laitteella on edeltäkäsin täytynyt olla pääsy siihen tietojaksoon tai niihin tietojaksoihin, joista nämä tekijät luetaan. Kun sitten laskutoimitus on suoritettu, tulo täytyy palauttaa määrättyyn paikkaan muistissa, josta se voidaan tulostaa tulostuslaitteella.

Mikä tahansa laskutoimitus sisältää sen tähden seuraavat toiminnat: lukeminen, tekijöiden sijoittaminen muistiin, tuloksen mahdollinen tarkistaminen, sen palauttaminen muistiin ja lopullisen tuloksen kirjoittaminen. Jopa yksinkertaisinkin laskutoimitus sisältää joukon suunniteltuja toimenpiteitä, jotka täytyy suorittaa, jos halutaan saattaa tehtävä loppuun suoritetuksi.

Kokonainen työsuoritus koostuu näistä yksityisistä askeleista, jotka on ryhmitelty sellaiseen järjestykseen, että ne ohjaavat konetta aikaansaamaan halutun lopputuloksen. Täten täytyy monimutkainen probleema ensin hajoittaa joukoksi koneen perustoimintoja ennen kuin se voidaan ratkaista. Jokainen näistä toiminnosta on koodattu muotoon, jota kone voi tulkita, ja sijoitettu muistiin tallennettuna ohjelmana.

Tallennetun ohjelman muuttamismahdollisuudet antavat tietojenkäsittelyjärjestelmälle miltei rajattoman joustavuuden. Yhtä ja samaa konetta voidaan käyttää erilaisten tehtävien suorittamiseen sijoittamalla muistiin kulloinkin käytettävä ohjelma. Mitä tahansa syöttölaitetta voidaan käyttää tähän tarkoitukseen, koska käskyt voidaan koodata konekielelle aivan kuin käsiteltävät tiedotkin.

Tallennettu ohjelma on helposti koneen käytettävissä ja se antaa koneelle mahdollisuuden muuttaa omaa ohjelmaansa toiminnan kestäessä esiintyvien tilanteiden mukaisesti. Tästä johtuen kone voi suorittaa rajoitetun asteista päättelyä sille mahdollisten toimintojen puitteissa.

Seuraavassa on esitetty lyhyt johdanto erityyppisiin ohjelmiin ja järjestelmiin (osituskäyttö, moniajo, jne.). Jäljempänä näitä seikkoja käsitellään vielä yksityiskohtaisemmin.

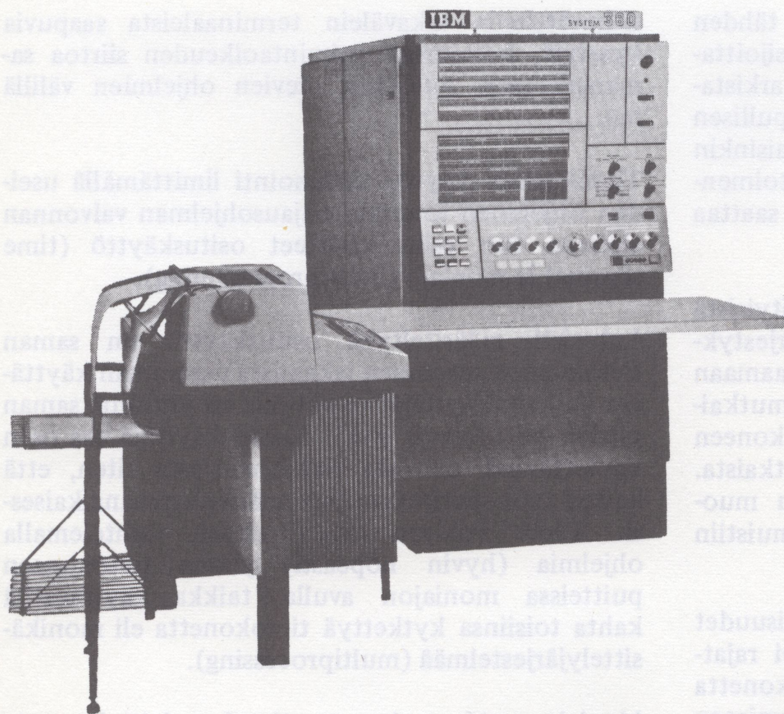
Tietojen kaukokäsittelyverkkojen ja monenlaisten tietokoneeseen kytkettävien syöttö- ja tulostuslaitteiden asianmukaisen toiminnan hoitamiseksi ovat sekä IBM että IBM-tietokoneiden käyttäjät kehittäneet tarvittavia ohjausohjelmia. Ohjausohjelmista on käytetty myös nimityksiä monitor-ohjelma ja valvontaohjelma (supervisor). Niiden tehtävänä, kuten nimikin sanoo, on toimia eräänlaisina 'liikenteen ohjaajina' muiden ohjelmien suhteen. Muut ohjelmat, probleema- tai käyttöohjelmat, hoitavat sovellutuskohtaisia tehtäviä ja luovuttavat sopivissa kohdissa, esim. tehtävän päätyttyä toimintaoikeuden ohjausohjelmalle, joka saattaa olla konstruoitu siten, että se pystyy käsittelemään säännöttömin aikavälein terminaaleista saapuvia kyselyjä, hoitamaan toimintaoikeuden siirtoa samanaikaisesti muistissa olevien ohjelmien välillä jne.

Tietokoneen käytön optimointi limittämällä useiden ohjelmien toiminta ohjausohjelman valvonnan alaisena tuo esiin käsitteet osituskäyttö (time sharing) ja moniajo (multiprogramming).

Lyhyesti määriteltynä osituskäyttö on saman tietokoneen resurssien jakamista useamman käyttäjän kesken (käyttäjät voivat olla eri yhtiöitä, saman yhtiön eri osastoja jne.). Kukin käyttäjä saa osan käytettävissä olevasta tietokoneajasta siten, että kaikki työt suoritetaan (näennäisen) samanaikaisesti. Tämä voidaan saada aikaan vaihtelemalla ohjelmia (hyvin nopeasti) yhden tietokoneen puitteissa moniajon avulla, taikka käyttämällä kahta toisiinsa kytkettyä tietokonetta eli monikäsittelyjärjestelmää (multiprocessing).

Moniajo voidaan kuvata tietokonelaitteiston ja valvontaohjelmien muodostamaksi järjestelmäksi, joka pystyy samanaikaisesti valvomaan yhden tai useamman käyttöohjelman toimintaa. Tämä tapahtuu limittämällä ohjelmien keskusyksikköön, muistiin ja syöttö- ja tulostuslaitteisiin kohdistuvat toiminnot. Sen vuoksi ohjausohjelman on pystyttävä identifioimaan se kohta missä kulloinkin toteutettavan ohjelman täytyy odottaa jonkin tapahtuman päättymistä. Siinä kohdassa ohjausohjelma aloittaa jonkin toisen tehtävän toteuttamisen. Tämän jälkeen ohjausohjelman on esim. pystyttävä palaamaan edelliseen ohjelmaan mikäli tämä on valmis jatkamaan toimintaansa. Koska muistissa saattaa samanaikaisesti olla useita keskeneräisiä ohjelmia, kelvollinen moniajojärjestelmä edellyttää tavallisesti sitä, että eri tehtäville on määritetty oma prioriteettinsa, eli järjestys, jossa niitä 'palvellaan'.

Osituskäyttö, moniajo ja monikäsittely ovat toiminnallisesti lähellä toisiaan ja niitä voidaan käyttää osituskäyttöperiaatteella, mutta jokin probleema saattaa vaatia useita tehtäviä, ajoja, joita moniajoon soveltuva laitteisto ja ohjelmointijärjestelmä voivat limittää. Samoin on mahdollista, että tietokoneen päätehtävänä on kaukokäsittelytoimintojen hoitaminen, ts. saapuvien ja lähtevien sanomien käsittely, mutta samanaikaisesti toteutetaan, sivutyönä, muita ohjelmia. Nämä kaksi esimerkkiä osituskäytön ja moniajon yhdistelymahdollisuuksista riittänevät tässä vaiheessa.



Kuva 9. Keskusyksikkö ja ohjauspöytä

Toimintayksiköt

Kaikki tietojenkäsittelyjärjestelmät koostuvat neljää lajia olevista yksiköistä: syöttölaitteet, tulostuslaitteet, muisti ja keskusyksikkö.

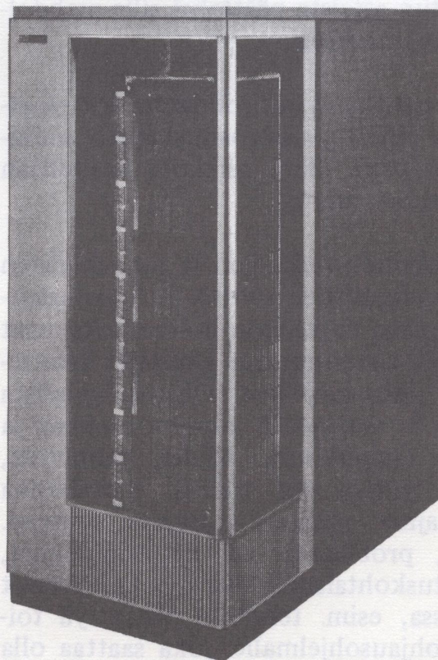
Keskusyksikkö

Keskusyksikkö (kuva 9) on koko tietojenkäsittelyjärjestelmän ohjauskeskus. Se voidaan jakaa kahteen osaan:

- Aritmeettis-looginen yksikkö
- Ohjausosa

Aritmeettis-looginen osa suorittaa sellaisia toimintoja kuin yhteenlasku, vähennyslasku, kertolasku, jakolasku, siirto, vertailu jne. Sillä on myös looginen kyky, kyky testata erilaisia ehtoja, jotka ilmaantuvat käsittelyn aikana, ja ryhtyä näiden tulosten vaatimiin toimenpiteisiin.

Ohjausosa ohjaa ja koordinoi koko tietojenkäsittelyjärjestelmän yksinkertaiseksi monikäyttöiseksi koneeksi. Nämä toiminnat sisältävät syöttö- ja tulostusyksikköjen ohjauksen ja keskusyksikön aritmeettisloogisen toiminnan valvonnan, tietojen siirtämisen muistiin ja sieltä pois koneen konstruktion puitteissa. Tämä osa ohjaa järjestelmää sen suunnittelijoilta peräisin olevien menetelmien mukaan.



Kuva 10. IBM 2361 suurmuisti

Muisti

Muisti on eräänlainen elektroninen rekisterikaappi, joka on varustettu täydellisellä hakemistolla ja on melkein silmänräpäyksellisesti koneen käytettävissä.

Kaikki tiedot täytyy sijoittaa muistiin ennen kuin kone voi käsitellä niitä. Syöttöyksikkö lukee tiedot muistiin, ja tiedot ovat sen jälkeen käytettävissä sisäiseen käsittelyyn. Jokainen muistipaikka, positio tai jakso on siten numeroitu, että kone voi tarvittaessa välittömästi paikantaa muistiin sijoitetun tiedon.

Kone voi järjestää muistissa olevat tiedot uudelleen lajittelemalla tai yhdistämällä erilaisia tietoja, joita se on ottanut vastaan eri syöttöyksiköiltä. Kone voi myös ottaa alkuperäiset tiedot muistista, laskea uusia tietoja ja sijoittaa tulokset takaisin muistiin.

Muistin koon eli kapasiteetin määrää se tietomäärä, jonka kone samanaikaisesti voi sisältää. Muutamissa koneissa muistikapasiteetti mitataan miljoonissa yksiköissä tai merkeissä, jotka muodostavat kokonaisia tietorekistereitä. Toisissa järjestelmissä muisti on pienempi ja tiedot pidetään siellä vain sen ajan, jolloin niitä käsitellään.

Näinollen muistin kapasiteettiin ja suunnitteluun vaikuttaa menetelmä, jolla järjestelmä käsittelee tietoja.

Systeemin/360 keskusmuisti koostuu seuraavista komponenteista: tietomuisti, eli se komponentti, josta yleensä käytetään nimitystä keskusmuisti, ja jonka kapasiteetti vaihtelee keskusyksikön mallista riippuen 4096 tavusta yli miljoonaan tavuun. Ohjausmuisti (control storage), joka yleensä sisältää erityisiä painetuiksi piireiksi rakennettuja mikro-ohjelmia tietokoneen sisäisten toimintojen ohjausta varten. Työmuisti (local storage), joka sisältää mm. yleis- ja eksponenttirekisterit. Suurmuisti (kuva 10), joita voidaan 1 miljoonan tavun elementteinä liittää Systeemiin/360, tosin vain suurempiin malleihin, 8 kappaletta ja siten lisätä koneen muistikapasiteettia 8 miljoonalla tavulla.

Edellisten lisäksi voidaan muistina käsitellä poimintamuistilaitteita, joihin tullaan tämän kirjan puitteissa palaamaan vielä myöhemmin. Poimintamuisti- ja magneettinauhayksiköistä voidaan käyttää yhteisnimitystä ulkoinen muisti tai apumuisti.

Muistin rakenne on sellainen, että tietoa voidaan tallentaa sinne monissa eri muodoissa - kokonaisina tietueina, tietueiden osina, numeroina, tavuina jne. Muistin kapasiteetti ilmaistaan tavallisesti merkkeinä (characters). Merkillä tarkoitetaan kirjaimia, numeroita ja erikoismerkkejä (välimerkkejä tms.). Systeemissä/360 'merkin' sijasta käytetään termiä 'tavu' (byte). Numerot voidaan esittää myös pakatussa muodossa ts. kaksi numeroa sopii samaan tilaan minkä yksi kirjain tai erikoismerkki vaatii.

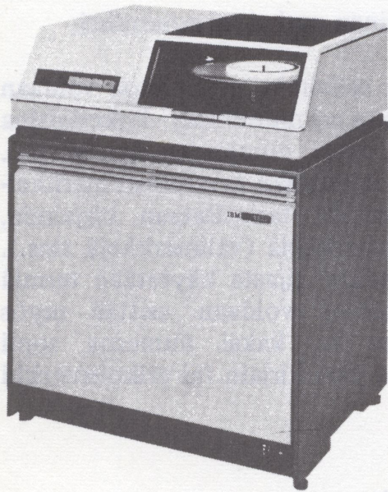
Syöttö- ja tulostuslaitteet

Tietojenkäsittelyjärjestelmä tarvitsee tehtäväänsä suorittaessaan välttämättä laitteet, jotka voivat sijoittaa tiedot järjestelmään ja antaa ne sieltä ulos. Suoraan järjestelmään liitetyt syöttö- ja tulostuslaitteet hoitavat näiden tehtävien suorittamisen (kuva 11).

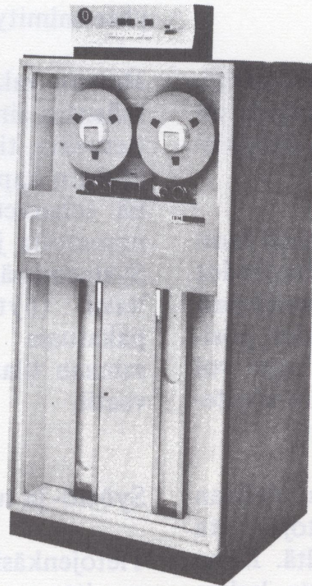
Syöttölaitteet lukevat eli tuntevat määrätylle tietojenvälitysmateriaalille koodatut tiedot ja saattavat nämä tiedot koneen käyttöön. Luettavat tiedot merkitään reikäkorteille ja paperinauhalle lävistettyinä reikinä, magneettinauhalle magnetisoituina pisteinä pitkin nauhaa, paperiasiakirjoissa magneettisella musteella painettuina kirjaimina, taikka piirroksina tai merkkiriveinä valokynän ja näyttölaitteen näppäimistön avulla.

Menetelmiin, joilla rekisteröidään konetta varten, sekä tietojen tallennusvälineiden ominaisuuksiin palataan tarkemmin myöhemmissä kappaleissa.

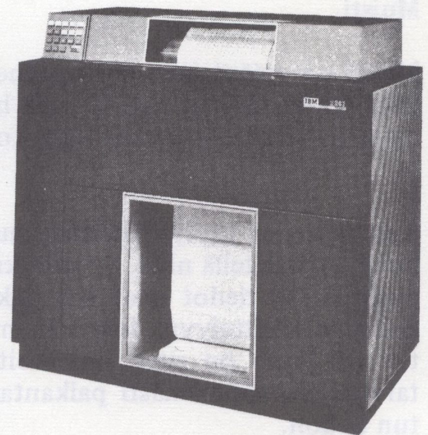
Tulostuslaitteet merkitsevät tai kirjoittavat tiedot koneesta reikäkorteille, reikänauhalle, magneettinauhalle tai paperille painettuina. Syöttö- ja tulostuslaitteiden määrä ja laatu riippuvat järjestelmän suunnittelusta ja sen käytöstä. Erityisiä tietojen muuntamisvaiheita, joissa jäljennetään toiselle materiaalille merkityt tiedot toiselle, liitetään kaikkiin tietokoneisiin. Esim. reikäkorteille merkityt tiedot voidaan automaattisesti sijoittaa magneettinauhalle. Tämä toiminta voi tapahtua ON-LINE eli hyväksikäyttäen keskusyksikköä tai OFF-LINE eli käyttämällä erillisiä syöttö- ja tulostuslaitteita.



IBM 2311 Levymuisti



IBM 2401 Magneettinauhayksikkö



IBM 1403 Rivikirjoitin



IBM 2260 Näyttölaite



IBM 2540 Kortinlukija-lävistin



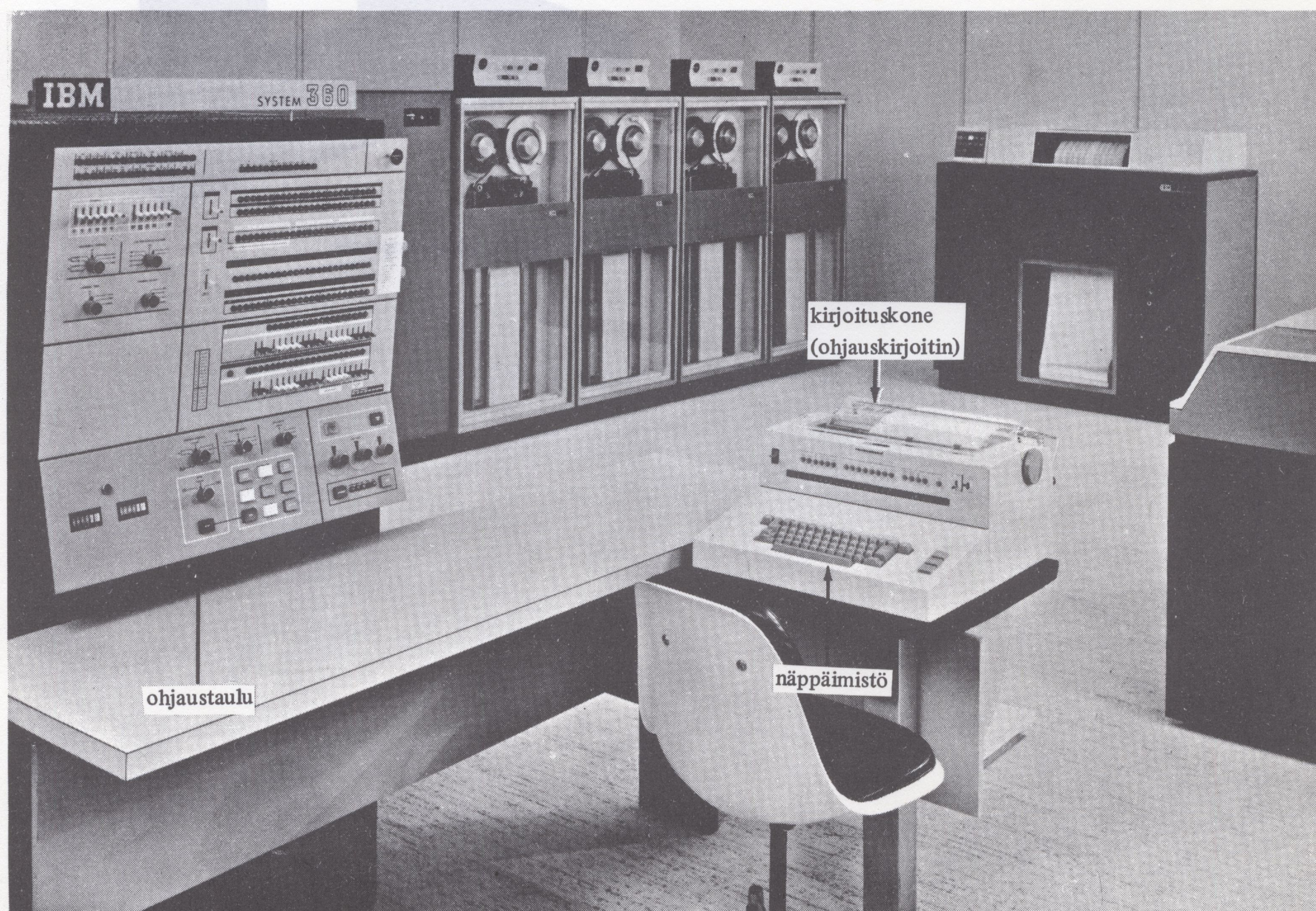
IBM 1009 Tietojensiirtoyksikkö

Kuva 11. Syöttö/tulostuslaitteita

Ohjauspöytä

Ohjauspöytä (kuva 12) muodostaa tietojenkäsittelyjärjestelmän ulkoisen ohjauksen. Kytkimillä voidaan kääntää virta päälle tai pois, aloittaa toiminta tai pysäyttää se, ohjata järjestelmän eri laitteita jne. Tietoja voidaan lukea sisään näppäimiä painamalla. Ohjauspöytään on asennettu valot siten, että niiden avulla voidaan nähdä koneessa olevat tiedot.

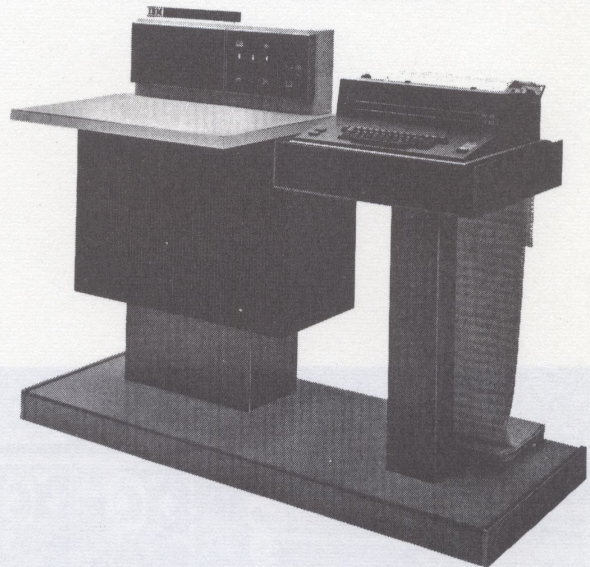
Eräissä järjestelmissä voidaan ohjauspöydällä olevaa kirjoituskonetta käyttää rajoitetusti tulostusyksikkönä. Se voi antaa kirjoittamalla tiedon käsittelyn päättymisestä tai sattuneesta virheestä. Se voi myös kirjoittaa lopputuloksia tai muita tietoja, jotka tekevät koneen käyttäjälle mahdolliseksi koneen toiminnan tarkkailun ja valvonnan. Toisaalta sitä voidaan käyttää syöttölaitteena - esim. muistissa olevan ohjelman muuttamiseksi - tietenkin sillä



Kuva 12. IBM Systeemin/360 mallin 40 ohjaustaulu, ohjauskirjoitin ja näppäimistö.

edellytyksellä, että ohjelmissa on tällaiseen varaudutta.

Ulkoisen ohjauspöydän (kuva 13) avulla saavutetaan suuri tehokkuus ja joustavuus tietokoneen ohjaamisessa sellaisissa tapauksissa, joissa tarvitaan kaksoisohjausta käsittely-yksikön ulkopuolella olevasta ohjausasemasta käsin.



Kuva 13. IBM 2150 ohjauspöytä

TIETOJEN ESITTÄMINEN

Symbolit toimivat tiedon välittäjinä, itse symboli ei ole tietoa, mutta se esittää sitä. Tämän sivun kirjoitusmerkit ovat symboleja ja ilmoittavat niitä ymmärtävälle lukijalle kirjoittajan tarkoituksen.

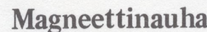

Kuva 14. Kommunikaatiosymboleita

Symbolien tarkoitus on tiedon esittäminen. Yhdelle voi symboli merkitä sitä, toiselle tätä, jollekin, joka ei sitä tunne, se ei merkitse kerrassaan mitään (kuva 14).

Tietojen antaminen tietokoneelle on monessa suhteessa samanlaista kuin asioiden kirjallinen ilmaisu jollekin henkilölle. Esitettävä tieto täytyy muuntaa symbolijoukoksi. Kielessämme näitä symboleja ovat aakkoset, numerot ja erikoismerkit. Näitä merkkejä kirjoitetaan paperille edeltä käsin määrättyyn järjestykseen ja lähetetään toiselle henkilölle, joka lukee ja tulkitsee ne.

Samalla tavalla täytyy tietokoneelle annettava tieto muuttaa symbolijoukoksi, jonka tietoja käsittelevä kone pystyy lukemaan ja ymmärtämään. Nämä symbolit eroavat ihmisten käyttämistä symboleista, sillä niiden täytyy olla koneen rakenteeseen ja toimintatapaan sopivia. Tietokoneen suunnittelija voi määrätä, millaisia symboleita käytetään ja mikä niiden tarkoitus on. Tärkeätä on vain se, että niillä voidaan esittää tietoa, toisin sanoen niistä muodostetaan kieli ihmisten ja koneiden välisiä toimintoja varten.


Reikäkortti


Magneettinauha

Enter partial payment below

MUNICIPAL WATER WORKS

Account Number	Gross Amount	Net Amount	Last Day To Pay Net
RL 45332	56 01	45 98	4 30 6-

DISCOUNT TERMS: 10 DAYS

Present Reading	Previous Reading	Consumption Gals.
3255686	2369014	887

E D JONES
745 CHESTNUT ST
ANYTOWN USA

PLEASE RETURN THIS WITH YOUR PAYMENT

Optisesti luettavia merkkejä


Reikänauha

YOUR NATIONAL BANK

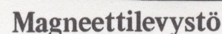
New York, N.Y.

SAMPLE VOID

DOLLARS

A. B. Depositor
Mary F. Depositor

Magneettimustemerkkejä


Magneettilevystö

Kuva 15. Tietojen tallennusvälineitä

Tietokoneissa käsiteltävät tiedot voivat olla tallennettuina reikäkorteille, reikänauhalle, magneettinauhalle, poimintamuistilaitteille, lomakkeille magneettisesti tai optisesti luettavina merkkeinä, mikrofilmille jne. Tämä luettelo kasvaa vuosi vuodelta. Joitakin tietojentallennusvälineitä on esitetty kuvassa 15.

Reikäkorteissa tiedot esitetään pienillä nelikulmaisilla rei'illä, jotka sijoitetaan korttiin määrättyihin paikkoihin. Samalla tavalla esittävät reikänauhan pienet pyöreät reiät tietoa. Magneettinauhalla tai levyllä symbolit ovat pieniä magnetoituja alueita, pisteitä, jotka on järjestetty määrätyn järjestelmän mukaan. Magneettiset kirjoitusmerkit kirjoitetaan paperille. Merkkien muoto ja musteen magneettiset ominaisuudet tekevät kirjoitetun tiedon lukemisen mahdolliseksi sekä koneelle että ihmisille. Optisesti luettavia kirjoitusmerkkejä voi lukea sekä kone että ihminen niiden muodon ja taustapaperin vaikutuksen vuoksi.

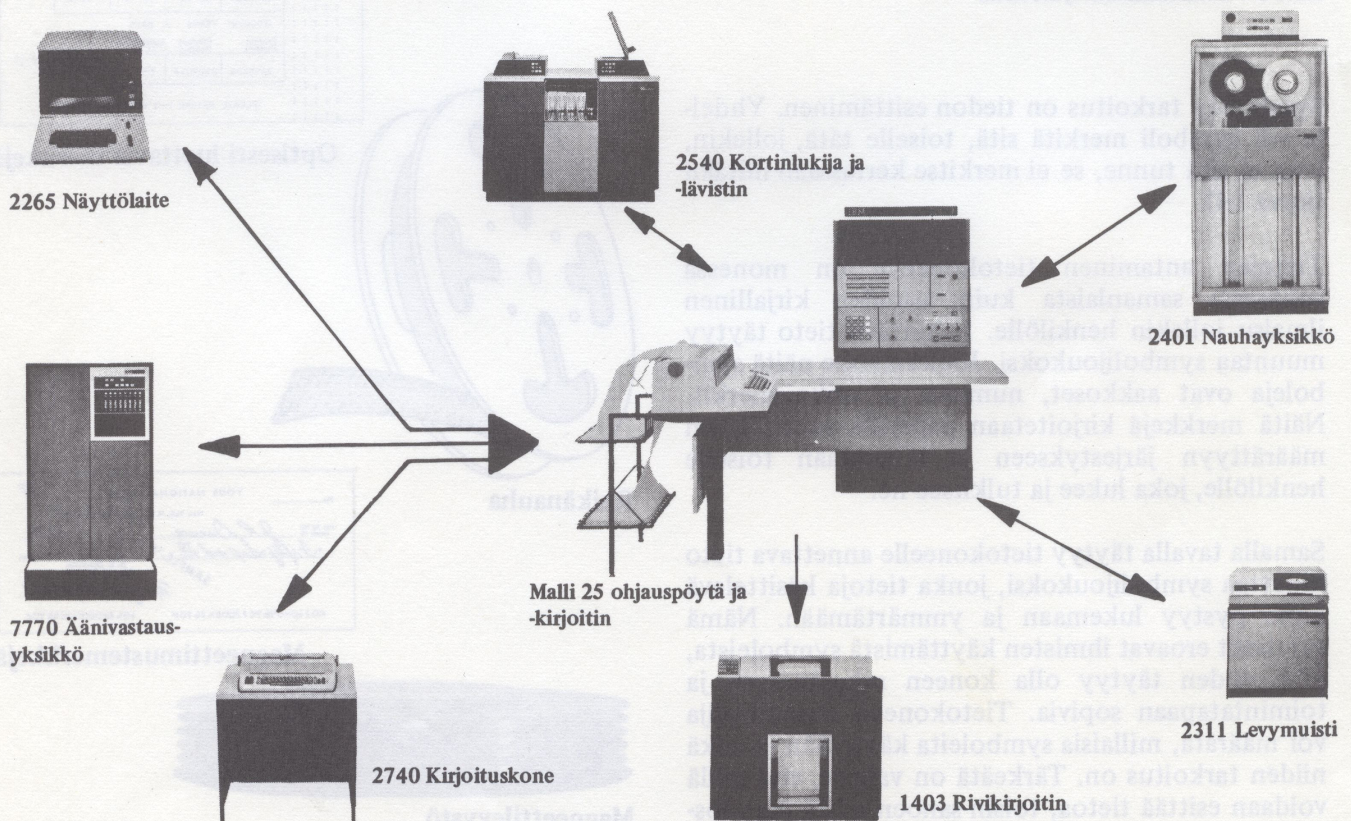
Jokainen tiedonesitystapa vaatii koodin tai erikoisen symbolijärjestelmän. Näitä koodeja kuvataan myöhemmin.

Tietokonejärjestelmän syöttölaite on laite, joka on rakennettu tunnistamaan eli lukemaan tiedot tiedontallennusvälineestä. Luettaessa tallennettu tieto muutetaan sähköiseen muotoon, toisin sanoen se esitetään sähköisillä merkeillä. Kone voi sitten käyttää tietoa suorittaessaan tietojenkäsittelytoimintaa. Tulostusyksikkö on laite, joka ottaa vastaan tiedot tietokoneelta ja tulostaa ne ohjelmassa määritetyn laitteen avulla.

Kaikkia syöttö/tulostuslaitteita ei voida käyttää suoraan kaikkien tietokoneiden kanssa. Kuitenkin johonkin tallennusvälineeseen sijoitettu tieto voidaan siirtää toiseen tallennusvälineeseen käytettäväksi toisessa järjestelmässä.

Esimerkiksi reikäkorteilla oleva tieto voidaan siirtää magneettinauhalle. Samoin magneettinauhalla oleva tieto voidaan siirtää reikäkorteille tai paperinauhalle tai painaa paperille.

Kuten ihminen voi antaa tiedonantoja koneelle, voi myös kone antaa niitä toiselle (kuva 16). Tämä koneitten välinen toiminta voi olla suoranaista tietojen vaihtoa sähköisessä muodossa lankojen,



Kuva 16. Koneiden välinen kommunikaatio

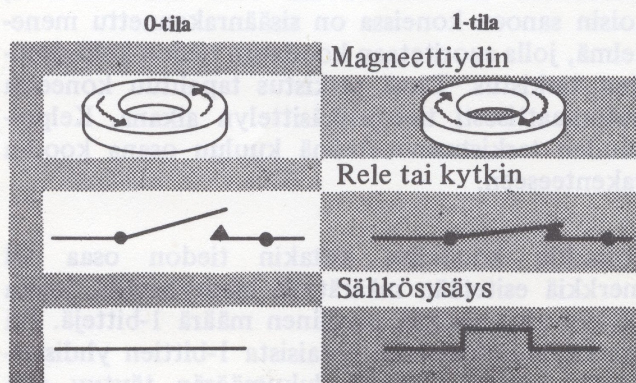
kaapeleiden tai radioaaltojen välityksellä tai yhden koneen tai järjestelmän tulostusta voidaan käyttää toisen koneen tai järjestelmän syöttötietona.

TIEDON ESITYS TIETOKONEESSA

Paitsi menetelmää tietojen esittämiseksi reikäkorteilla, paperinauhalla, magneettinauhalla ja magneettisina merkkeinä, täytyy olla myös menetelmä tietojen esittämiseksi koneen sisällä. Tietokoneessa tietoa edustavat monet sähköiset komponentit, transistorit, magneettirenkaat, langat jne. Tietojen varastoimista ja kulkua edustavat sähköiset signaalit ja merkit. Näiden signaalien olemassaolo tai puuttuminen määrätyissä virtapiireissä, samoin kuin reikien esiintyminen tai puuttuminen reikäkorteissa, on tiedonesittämismenetelmä.

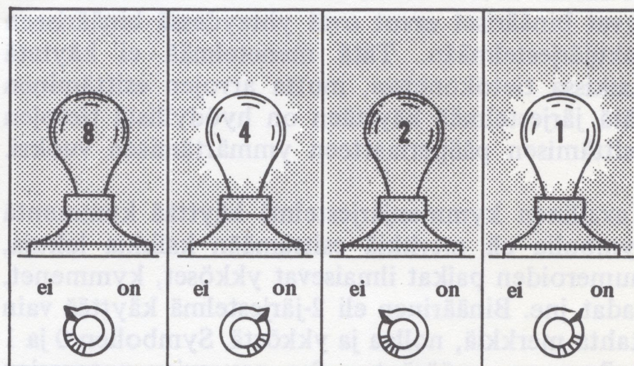
Binääriset tilat

Tietokoneen toiminta on aina tietyssä mielessä binääristä, toisin sanoen tietokoneen komponenteilla voi olla vain kaksi tilaa. Esimerkiksi tavallinen hehkulamppu toimii binäärisellä tavalla: se joko valaisee tai ei valaise. Valon olemassaolo tai sen puuttuminen osoittaa, johdetaanko lamppuun virtaa vai ei. Samalla tavalla tietokoneessa transistorit joko johtavat tai eivät johda. Magneettinen aine voidaan magnetisoida kahteen vastakkaiseen suuntaan ja määrätyt jännitteet joko ovat olemassa tai puuttuvat (kuva 17). Komponenttien binääriset toiminnat ovat merkkejä koneelle, kuten sähkölampun valon olemassaolo tai puuttuminen on merkki ihmiselle.



Kuva 17. Binäärisiä komponentteja

Tietojen esittämiseksi tietokoneessa annetaan binäärisille indikaattoreille tai indikaattoriryhmille määrätty arvo. Esimerkiksi yksikkö kymmenjärjestelmän lukujen esittämistä varten voitaisiin rakentaa neljästä lampusta ja katkaisijoista, joilla lamput voidaan sytyttää ja sammuttaa (kuva 18).



Kuva 18. Desimaalisen tiedon esitys binäärisillä kytkimillä

Lampuille annetaan arvot 1, 2, 4, 8. Kun lamppu valaisee, se esittää asianomaista lukua. Jos se ei valaise, lukua ei oteta huomioon. Sellaisessa järjestelmässä esitettävä luku on valaisevien lamppujen osoittamien lukujen summa. Täten voidaan esittää luvut 0-15. Lukua 0 esitettäessä kaikki lamput ovat pimeinä, lukua 15 esitettäessä kaikki valaisevat. Luku 9 esitetään siten, että lamput 8 ja 1 valaisevat ja lamput 4 ja 2 ovat pimeinä, luku 5 esitetään siten, että 1 ja 4 valaisevat, 8 ja 2 ovat pimeinä jne.

Esimerkissä käytetyt lamppujen arvot voisivat olla jotkut muut. Tällainen vaihto toisi mukanaan uusien arvojen määräämisen ja toimintakaavan päättämisen. Tietokoneessa arvot, jotka määrätty binääristen indikaattoreiden joukko esittää, muodostavat tietojen esittämiseen tarvittavan koodin eli kielen. Koska binääriset indikaattorit esittävät tietoa tietokoneessa, binääristä merkitsemistapaa käytetään havainnollistamaan näitä indikaattoreita. Binäärinen merkintätapa käyttää vain kahta symbolia, 0 ja 1, esittämään kaikkia lukuja. Jossakin binääripositiossa 0 merkitsee asianomaisen annetun arvon puuttumista ja 1 merkitsee sen olemassaoloa. Esimerkiksi kuvassa 18 esitettyjä lamppuja voidaan kuvata binääriluvulla 0101.

Binäärisiä merkkejä 0 ja 1 kutsutaan nimellä bitti. 0 merkitsee, ettei bittiä ole ja 1 merkitsee että se on. Vaikka sekä 0- että 1-bitti ovat välttämättömiä binääristen merkkien tai merkkiryhmien esittämiseen, tarkoitetaan nimityksellä bitti yleensä bittiä

1. Esimerkiksi kuvassa 18 voitaisiin sanoa, että bitti on paikoissa 1 ja 4.

Binääriset lukujärjestelmät

Joissakin tietokoneissa binäärisiin merkkeihin kuuluvat binääriset arvot ovat yhteydessä binääriseen lukujärjestelmään. Tätä menetelmää ei käytetä kaikissa tietokoneissa, mutta arvojen esittäminen tätä järjestelmää käyttäen on hyödyllistä tietojen esittämisen pääperiaatteen ymmärtämisen vuoksi.

Tavallinen kymmenjärjestelmä käyttää kymmentä symbolia eli merkkiä esittämään kaikkia lukuja, numeroiden paikat ilmaisevat ykköset, kymmenet, sadat jne. Binäärinen eli 2-järjestelmä käyttää vain kahta merkkiä, nollaa ja ykköstä. Symbolien 0 ja 1 paikan arvo määräytyy 2:n nousevien potenssien mukaan, siis n:s paikka oikealta on arvoltaan 2^{n-1}, siten ensimmäinen paikka merkitsee 1:tä, toinen 2:ta, kolmas 4:ää, neljäs 8:aa jne. (kuva 19).

8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1

Kuva 19. Binäärinumeroiden sijainnolliset arvot

Puhtaassa 2-järjestelmässä binääriset merkit ilmoittavat jokaisessa paikassa kakkosen potenssin olemassaolon tai puuttumisen. Bitti 1 merkitsee kakkosen potenssin olemassaoloa ja bitti 0 sen puuttumista. Paikkojen arvo ei siis merkitse ykkösiä, kymmeniä, satoja jne., vaan ykkösiä, kakkosia, nelosia, kahdeksikkoja jne. Esimerkiksi luku 12 tätä järjestelmää käyttäen esitetään symboleilla 1100, joka tarkoittaa 8+4+0+0 = 12.

Desimaali arvo	Position arvo 16	8	4	2	1
0	0	0	0	0	0
1	0	0	0	0	1
2	0	0	0	1	0
3	0	0	0	1	1
4	0	0	1	0	0
5	0	0	1	0	1
6	0	0	1	1	0
7	0	0	1	1	1
8	0	1	0	0	0
9	0	1	0	0	1
10	0	1	0	1	0
11	0	1	0	1	1
12	0	1	1	0	0
13	0	1	1	0	1
14	0	1	1	1	0
15	0	1	1	1	1
16	1	0	0	0	0

Kuva 20. Desimaalilukujen 0-16 binäärinen esitysmuoto

Kuva 20 esittää 2-järjestelmän luvut 0-16. Huomattava on, että luvut 0-9 voidaan esittää neljällä 2-järjestelmän merkillä. Kymmenjärjestelmän numeroiden esittämistä 2-järjestelmän avulla kutsutaan nimellä BINARY CODED DECIMAL, josta käytetään lyhennystä BCD. Kuvassa 21 esitetään kymmenjärjestelmän luku 265.498 BCD-muodossa.

Desim. numero	2				6				5				4				9				8			
Binäärinen arvo	0	0	1	0	0	1	1	0	0	1	0	1	0	1	0	0	1	0	0	1	1	0	0	0
Position arvo	8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1

Kuva 21. Desimaaliluku 265498 binäärisesti koodattuna

TIETOKONEKOODIT

Tietojen esittäminen tapahtuu koodijärjestelmän avulla. Tietokoneessa tämä merkitsee sitä, että tietoalkiota edustaa tietyn järjestelmän mukainen symboli, joukko binääri-indikaattoreita. Numeroiden ja kirjaimien esittämiseen voidaan esim. käyttää 8 binääri-indikaattorin eli 8 bitin järjestelmää. Näitä kahdeksaa bittiä (joilla voi kullakin olla arvo 0 tai 1) sopivasti järjestelemällä voidaan eri bittikombinaatioilla esittää kaikki merkit.

Eräitä käytössä olevia koodeja ovat 6 bitin aakkosnumeerinen koodi, 8 bitin aakkosnumeerinen koodi, 2/5 (kaksi-viidestä) koodi ja 6 bitin numeerinen koodi.

Koodin tarkistus

Useimmat tietokonekoodit ovat itsetarkistavia, toisin sanoen koneissa on sisäänrakennettu menetelmä, jolla suoritetaan koodatun tiedon kelpoisuuden tarkistus. Tämä tarkistus tapahtuu koneessa automaattisesti tietojenkäsittelyn aikana. Kelpoisuuden tarkistusmenetelmä kuuluu osana koodin rakenteeseen.

Joissakin koodeissa kutakin tiedon osaa tai merkkiä esitetään määrätyllä bittiryhmällä, jonka täytyy aina sisältää parillinen määrä 1-bittejä. Eri merkit muodostetaan erilaisista 1-bittien yhdistelmistä, mutta 1-bittien lukumäärän täytyy olla jokaisessa kelvollisessa merkissä parillinen. Koodijärjestelmässä huomataan heti merkki, jossa on

pariton määrä 1-bittejä, ja tietokone osoittaa virhettä. Samoin voidaan myös käyttää sellaista koodia, jossa 1-bittien lukumäärän täytyy olla pariton. Kone ilmoittaa virheen, kun esiintyy merkki, jossa on parillinen määrä 1-bittejä.

Tällainen tarkistus tunnetaan nimellä pariteettitarkistus. Parillista 1-bittien lukumäärää käyttävillä koneilla sanotaan olevan parillinen pariteetti ja paritonta 1-bittien lukumäärää käyttävillä koneilla sanotaan vastaavasti olevan pariton pariteetti.

Joissakin koodeissa ykkösbittien lukumäärä per merkki on kiinteä. Esimerkiksi merkkien esittämisessä käytetään 8 bittiä, mutta kussakin merkissä voi ykkösbittejä olla vain 4. Merkit joissa ykkösbittien määrä poikkeaa neljästä aiheuttavat virheen. Tällaista tarkistusmenetelmää nimitetään lukumäärätarkistukseksi ja sitä käytetään enimmäkseen tietojen kaukosiirrossa.

Kuuden bitin aakkosnumeerinen koodi
(BCD eli Binary Coded Decimal System)

Tällä koodilla kaikki merkit - numerot, aakkoset ja erikoismerkit - esitetään kuudella bitillä (seitsemäs on tarkistus- eli pariteettibitti). Bittipositiot voidaan jakaa kolmeen ryhmään: tarkistusbitti, 2 vyöhykebittiä, ja 4 numerobittiä (kuva 22).

Tarkist bitti	Vyöhykebitit		Numerobitit			
C	B	A	8	4	2	1

Kuva 22. 6-bitin aakkosnumeerisen koodin bittipaikat

Neljälle numerobitille on annettu arvot 8, 4, 2 ja 1 ja niillä esitetään BCD-muodossa numeroita 0-9 (kuva 23). On huomattava, että nolla esitetään yhdistelmällä 1010, mikä on oikeastaan luku 10 binäärisessä järjestelmässä. Numeroiden 0-9 esittämisessä ei käytetä vyöhykebittejä B ja A.

Vyöhyke- ja numerobittien yhdistelmillä esitetään kirjaimet ja erikoismerkit. B- ja A-biteistä voidaan saada yhdistelmät 10, 01 ja 11.

C-tarkistusbitti on ainoastaan tarkistusta varten. Koska kuuden bitin koodi on parillinen koodi, täytyy merkkejä esitettäessä 1-bittien lukumäärän olla jokaisessa merkissä parillinen, muuten merkki on virheellinen. Tarkistusbitti on magnetoitu silloin, kun merkin 1-bittien lukumäärä on pariton.

Tarkistukseen tulevat mukaan sekä vyöhyke- että numerobitit. Jos merkin 1-bittien lukumäärä on parillinen ilman C-bittiä, C-bitti on 0.

Desim. numero	Position arvo 8	4	2	1
0	1	0	1	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Kuva 23. Desimaalilukujen 0-9 bittiyhdistelmät, 6-bitin aakkosnumeerinen koodi

Standardi BCD-koodi

Tietojen yhteensopivuuden ja vaihtokelpoisuuden aikaansaamiseksi eri tietokonejärjestelmien välillä kehitettiin ns. standardi BCD-koodi. Se sisältää 64 erilaista merkkiä.

Kuvassa 24 esitetään BCD-koodi, jota käytetään tietojenkäsittelyjärjestelmissä 1401, 1410, 7010, 7040 ja 7044. Kuvassa on jokaisen 64 eri bittiyhdistelmän keskinäinen suuruusjärjestys, graafinen merkki, korttikoodi ja BCD-koodi. Pariteettitarkistukseen käytetyn C-bitin merkitys riippuu tietokoneesta. Jos järjestelmä käyttää paritonta pariteettia, C-bitti magnetoidaan automaattisesti niihin merkkeihin, joissa 1-bittien lukumäärä on parillinen. Samoin jos järjestelmä käyttää parillista pariteettia, C-bitti magnetoidaan kaikkiin parittoman 1-bittien määrän sisältäviin merkkeihin.

Viisi bittiyhdistelmää kirjoittuu kahtena merkkinä, rivikirjoittimen merkkivalikoimasta riippuen. Vaihtoehtoja nimitetään merkkivalikoima 1:ksi ja 2:ksi (kuva 25).

Merkkivalikoimaa 1 käytetään yleensä raporttien kirjoittamisessa ja muussa yleisessä käytössä, kun taas valikoimaa 2 käytetään kehittyneissä ohjelmointikielissä (FORTRAN, COBOL) matemaattisina symboleina.

Kuvassa 26 esitetään BCD-koodiin sisältyvien erikoismerkkien standardisoidut nimet.

Merkki Report	Merkki Program	Korttikoodi	BCD-koodi						
b		No Punches	C						
.		12-3-8		B	A	8		2	1
◻	)	12-4-8	C	B	A	8	4		
[		12-5-8		B	A	8	4		1
<		12-6-8		B	A	8	4	2	
≢		12-7-8	C	B	A	8	4	2	1
&	+	12	C	B	A				
$		11-3-8	C	B		8		2	1
*		11-4-8		B		8	4		
]		11-5-8	C	B		8	4		1
;		11-6-8	C	B		8	4	2	
Δ		11-7-8		B		8	4	2	1
–		11		B					
/		0-1	C		A				1
,		0-3-8	C		A	8		2	1
%	(	0-4-8			A	8	4		
^		0-5-8	C		A	8	4		1
\		0-6-8	C		A	8	4	2	
⧻		0-7-8			A	8	4	2	1
ƀ		2-8			A				
#	=	3-8				8		2	1
@	'	4-8	C			8	4		
:		5-8				8	4		1
>		6-8				8	4	2	
√		7-8	C			8	4	2	1
?		12-0	C	B	A	8		2	
A		12-1		B	A				1
B		12-2		B	A			2	
C		12-3	C	B	A			2	1
D		12-4		B	A		4		
E		12-5	C	B	A		4		1
F		12-6	C	B	A		4	2	
G		12-7		B	A		4	2	1
H		12-8		B	A	8			
I		12-9	C	B	A	8			1
!		11-0		B		8		2	
J		11-1	C	B					1
K		11-2	C	B				2	
L		11-3		B				2	1
M		11-4	C	B			4		
N		11-5		B			4		1
O		11-6		B			4	2	
P		11-7	C	B			4	2	1
Q		11-8	C	B		8			
R		11-9		B		8			1
‡		0-2-8			A	8		2	
S		0-2	C		A			2	
T		0-3			A			2	1
U		0-4	C		A		4		
V		0-5			A		4		1
W		0-6			A		4	2	
X		0-7	C		A		4	2	1
Y		0-8	C		A	8			
Z		0-9			A	8			1
Ø		0	C			8		2	
1		1							1
2		2						2	
3		3	C					2	1
4		4					4		
5		5	C				4		1
6		6	C				4	2	
7		7					4	2	1
8		8				8			
9		9	C			8			1

Low→ ... High→ Verroitusjärjestys

NOTE: Tape may use even parity.

Kuva 24. Standardi BCD-koodi

BCD Code	Graphic Subset 1 Print Arrangement A	Graphic Subset 2 Print Arrangement H
8-2-1	#	=
8-4	@	'
A-8-4	%	(
B-A	&	+
B-A-8-4	◻	)

Kuva 25. Erikoismerkkivalikoimat 1 ja 2

SYMBOL	NAME
≢	Group Mark
‡	Record Mark
⧻	Segment Mark
^	Word Separator
@	At Sign
#	Number Sign
&	Ampersand
+	Plus
*	Asterisk
%	Percent
/	Slash
\	Backslash
◻	Lozenge
b	Blank
ƀ	Substitute Blank
(	Left Parenthesis
)	Right Parenthesis
[	Left Bracket
]	Right Bracket
√	Tape Mark
<	Less than
>	Greater than
=	Equal to
;	Semicolon
:	Colon
.	Period or Point
'	Prime or Apostrophe
-	Minus or Hyphen (Dash)
Δ	Delta

Kuva 26. Erikoismerkit

Kahdeksan bitin aakkosnumeerinen koodi (EBCDIC)

Tässä järjestelmässä (kuva 27) kukin merkki sisältää 8 bittiä sekä tarkistus- eli pariteettibitin. Kahdeksalla bitillä voidaan esittää 256 erilaista merkkiä. Tämän koodin avulla voidaan esittää sekä

EBCDIC	Bit Configuration
NUL	0000 0000
SOH	0000 0001
STX	0000 0010
ETX	0000 0011
PF	0000 0100
HT	0000 0101
LC	0000 0110
DEL	0000 0111
	0000 1000
RLF	0000 1001
SMM	0000 1010
VT	0000 1011
FF	0000 1100
CR	0000 1101
SO	0000 1110
SI	0000 1111
DLE	0001 0000
DC1	0001 0001
DC2	0001 0010
TM	0001 0011
RES	0001 0100
NL	0001 0101
BS	0001 0110
IL	0001 0111
CAN	0001 1000
EM	0001 1001
CC	0001 1010
CU1	0001 1011
IFS	0001 1100
IGS	0001 1101
IRS	0001 1110
IUS	0001 1111
DS	0010 0000
SOS	0010 0001
FS	0010 0010
	0010 0011
BYP	0010 0100
LF	0010 0101
ETB	0010 0110
ESC	0010 0111
	0010 1000
	0010 1001
SM	0010 1010
CU2	0010 1011
	0010 1100
ENQ	0010 1101
ACK	0010 1110
BEL	0010 1111
	0011 0000
	0011 0001
SYN	0011 0010
	0011 0011
PN	0011 0100
RS	0011 0101
UC	0011 0110
EOT	0011 0111
	0011 1000
	0011 1001
	0011 1010
CU3	0011 1011
DC4	0011 1100
NAK	0011 1101
	0011 1110
SUB	0011 1111
SP	0100 0000
	0100 0001
	0100 0010
	0100 0011
	0100 0100

EBCDIC	Bit Configuration
	0100 0101
	0100 0110
	0100 0111
	0100 1000
	0100 1001
¢ [	0100 1010
.	0100 1011
<	0100 1100
(	0100 1101
+	0100 1110
|	0100 1111
&	0101 0000
	0101 0001
	0101 0010
	0101 0011
	0101 0100
	0101 0101
	0101 0110
	0101 0111
	0101 1000
	0101 1001
!]	0101 1010
$	0101 1011
*	0101 1100
)	0101 1101
;	0101 1110
¬	0101 1111
–	0110 0000
/	0110 0001
	0110 0010
	0110 0011
	0110 0100
	0110 0101
	0110 0110
	0110 0111
	0110 1000
	0110 1001
7/12	0110 1010
,	0110 1011
%	0110 1100
_	0110 1101
>	0110 1110
?	0110 1111
	0111 0000
	0111 0001
	0111 0010
	0111 0011
	0111 0100
	0111 0101
	0111 0110
	0111 0111
	0111 1000
6/0	0111 1001
:	0111 1010
#	0111 1011
@	0111 1100
'	0111 1101
=	0111 1110
"	0111 1111
	1000 0000
a	1000 0001
b	1000 0010
c	1000 0011
d	1000 0100
e	1000 0101
f	1000 0110
g	1000 0111
h	1000 1000
i	1000 1001

EBCDIC	Bit Configuration
	1000 1010
	1000 1011
	1000 1100
	1000 1101
	1000 1110
	1000 1111
	1001 0000
j	1001 0001
k	1001 0010
l	1001 0011
m	1001 0100
n	1001 0101
o	1001 0110
p	1001 0111
q	1001 1000
r	1001 1001
	1001 1010
	1001 1011
	1001 1100
	1001 1101
	1001 1110
	1001 1111
	1010 0000
—	1010 0001
s	1010 0010
t	1010 0011
u	1010 0100
v	1010 0101
w	1010 0110
x	1010 0111
y	1010 1000
z	1010 1001
	1010 1010
	1010 1011
	1010 1100
	1010 1101
	1010 1110
	1010 1111
	1011 0000
	1011 0001
	1011 0010
	1011 0011
	1011 0100
	1011 0101
	1011 0110
	1011 0111
	1011 1000
	1011 1001
	1011 1010
	1011 1011
	1011 1100
	1011 1101
	1011 1110
	1011 1111
PZ 7/11	1100 0000
A	1100 0001
B	1100 0010
C	1100 0011
D	1100 0100
E	1100 0101
F	1100 0110
G	1100 0111
H	1100 1000
I	1100 1001
	1100 1010
	1100 1011
⑀	1100 1100
	1100 1101
⑂	1100 1110

Kuva 27. Bittiyhdistelmät, EBCDIC

pienet että isot kirjaimet, enemmän erikoismerkkejä, joita tietyt syöttö- ja tulostuslaitteet vaativat. Tällä hetkellä joillakin bittikombinaatioilla ei ole mitään erityistä 'tehtävää' ts. ne eivät ole ohjausmerkkejä eikä niillä ole mitään graafista vastinetta. Ne ovat varalla tulevaisuutta varten. EBCDIC on toinen Systeemissä/360 käytetyistä koodijärjestelmistä.

EBCDIC	Bit Configuration
	1100 1111
MZ 7/13	1101 0000
J	1101 0001
K	1101 0010
L	1101 0011
M	1101 0100
N	1101 0101
O	1101 0110
P	1101 0111
Q	1101 1000
R	1101 1001
	1101 1010
	1101 1011
	1101 1100
	1101 1101
	1101 1110
	1101 1111
RM 5/12	1110 0000
	1110 0001
S	1110 0010
T	1110 0011
U	1110 0100
V	1110 0101
W	1110 0110
X	1110 0111
Y	1110 1000
Z	1110 1001
	1110 1010
	1110 1011
⑁	1110 1100
	1110 1101
	1110 1110
	1110 1111
0	1111 0000
1	1111 0001
2	1111 0010
3	1111 0011
4	1111 0100
5	1111 0101
6	1111 0110
7	1111 0111
8	1111 1000
9	1111 1001
⌇	1111 1010
	1111 1011
	1111 1100
	1111 1101
	1111 1110
EO	1111 1111

Kuva 27. Bittiyhdistelmät, EBCDIC (jatko)

Kahdeksan bitin aakkosnumeerinen standardikoodi (USASCII-8)

USASCII on tietoliikennelaitteiden ja tietokoneiden käyttäjien yhteistyönä kehittämä 7 bitin koodi. Tarkoituksena oli yksinkertaistaa ja standardisoida koneiden ja systeemien välistä kommunikaatiota.

Koska Systeemi/360 käyttää 8 bitin koodia, oli välttämätöntä laajentaa USASCII 8-bittiseksi. Tästä laajennetusta koodista käytetään lyhennystä USASCII-8. Koodia voidaan käyttää Systeemin/360 sisäisiin toimintoihin sekä syöttö- ja tulostustoiminnassa niillä välineillä, joita varten USASCII on standardisoitu.

LUKUJÄRJESTELMÄT

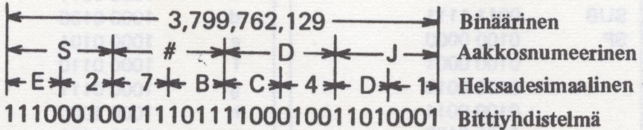
Binäärijärjestelmä

Binääristä tietojen esitystapaa käyttävistä tietokoneista tyypillisenä esimerkkinä on Systeemi/360.

Tiedon perusyksikkö on tavu. Neljä tavua muodostaa sanan - 32 peräkkäisen bitin tietojonon - jota käsitellään kokonaisuutena hieman samoin kuin merkkiä tai numeroa muissa järjestelmissä.

Binäärisen lukujärjestelmän mukaan biteillä sanan puitteissa on paikasta riippuva merkitys. Toisin sanoen bitin sijainti määrää sen lukuarvon. Bittien desimaaliset arvot (oikealta vasemmalle) ovat 1, 2, 4, 8, 16, 32, 64 jne. (ks. kuva 19).

Vaikka sanan bittien arvot ovatkin binäärijärjestelmän mukaiset, voidaan niitä tulkita tai käsitellä muillakin tavoilla kuin binääriluvun edellyttämällä. Esim. 32 bitin sanaa (kuva 28) voidaan käsitellä yhtenä binäärilukuna, kahdeksana heksadesimaalinumerona, neljänä aakkosnumeerisena merkkinä tms. ohjelmoijan määritysten mukaan.



Kuva 28. 32-bitin sana

Oktaalijärjestelmä

On selvää, että binääriluvut vaativat moninkertaisen määrän positioita vastaavien lukujen desimaaliesitykseen verrattuna. Suullisesti ja kirjallisesti esitettyinä binääriluvut ovat kömpelöitä. Pitkät jonot nollia ja ykkösiä lukuesityksissä tekisivät kaiken kommunikaation erittäin hankalaksi. On siis välttämätöntä käyttää jotakin 'pikakirjoitusmenetelmää', lyhyempää merkintätapaa. Lukujen muunto järjestelmästä toiseen on hyvin helppoa ja se johtuu niiden läheisestä suhteesta binäärijärjestelmään. Oktaalijärjestelmän kantalukuna on 8. Lukujärjestelmä sisältää siten 8 symbolia: 0, 1, 2, 3, 4, 5, 6 ja 7. Numerot 8 ja 9 puuttuvat. Suhde binäärijärjestelmään on yksinkertaisesti se, että 3 binääripositiota vastaa yhtä oktaalipositiota. Kuvan 29 taulukko selvittää binääri-, oktaali- ja desimaalijärjestelmien väliset suhteet sekä lukujen muunnon.

BINARY	OCTAL	DECIMAL
000	0	0
001	1	1
010	2	2
011	3	3
100	4	4
101	5	5
110	6	6
111	7	7

Tässä vaiheessa tapahtuu siirto seuraavaksi ylempään merkkipaikkaan, koska kaikki järjestelmän kahdeksan symbolia on jo käytetty.

BINARY	OCTAL	DECIMAL
001 000	10	8
001 001	11	9
001 010	12	10
001 011	13	11
001 100	14	12
.	.	.
.	.	.
.	.	.

Kuva 29. Binääri/oktaali/desimaalimuunto

On muistettava, että tietokoneen sisäinen logiikka sisältää vain binäärinumerot - nollan ja ykkösen. Oktaalijärjestelmä on vain eräs lyhyempi tapa lukea ja kirjoittaa binäärilukuja. Koska kantaluku on 8, oktaaliluvun numerot edustavat 8:n nousevien potenssien kertoimia.

Esim. oktaaliluku 173

$$= (1 \times 8^2) + (7 \times 8^1) + (3 \times 8^0)$$
$$= 64 + 56 + 3$$
$$= 123 \text{ (= luvun desimaalinen arvo).}$$

Muistamalla mitä numero binääri- tai oktaalijärjestelmässä edustaa, luku voidaan muuntaa desimaalimuotoon käyttämällä edellä esitettyä menetelmää. Suurten lukujen ollessa kyseessä menetelmä on kuitenkin hankala. Seuraavat jaksot käsittelevät hieman parempia lukujen muuntomenetelmiä.

KOKONAISLUKUJEN MUUNTAMINEN

Desimaaliluvun muunto oktaalimuotoon

Sääntö: Desimaaliluku jaetaan 8:lla niin monta kertaa että viimeinen jaettava on pienempi kuin 8. Kunkin jakolaskun jälkeen tallennetaan jakojäännös ja kun kaikki laskutoimitukset on suoritettu, oktaaliluku muodostetaan jakojäännöksistä.

Esim. Desimaaliluvun 149 muunto oktaalimuotoon

```
8 |149   Jäännös  5
8 |18      "      2   = 225
8 |2       "      2     ↑
   0
```

Ensin jaetaan alkuperäinen luku 8:lla. Tämän jakolaskun jakojäännös (5) vastaa oktaaliluvun 'ykkösiä'. Edelleen saatu osamäärä jaetaan 8:lla ja jakojäännös sijoitetaan kehitettävään oktaalilukuun (2). Samat toimenpiteet toistuvat kunnes jaettava eli edellisestä jakolaskusta saatu osamäärä on pienempi kuin jakaja. Viimeisestä osamäärästä tulee sitten oktaaliluvun korkein eli vasemmanpuoleisin numero.

Oktaaliluvun muunto desimaalimuotoon

Esimerkki: Oktaaliluvun 225 muunto.
Sääntö: Luvun ensimmäinen numero kerrotaan 8:lla, seuraava numero lisätään tuloon, saatu summa kerrotaan 8:lla jne. kunnes kaikki numerot on käytetty.

```
   2    2    5
 × 8
  16
 + 2 ←
  18
 × 8
 144
 + 5 ←
 149
```

Luvun vasemmanpuoleisin eli ensimmäinen numero (2) kerrotaan 8:lla ja tuloon lisätään seuraava numero (5). Kun luvun viimeinen numero on näin käytetty, prosessi päättyy ja laskutoimitusten tuloksena saatu luku on etsitty vastaus.

Oktaaliluvun muunto binääriseksi ja binääriluvun muunto oktaalimuotoon

Sääntö: Luku esitetään kolmen binäärinumeron ryhmissä.

Esim.

Oktaali - binääri

```
 2   2   5
010 010 101 = 010 010 101
```

Binääri - oktaali

```
010 010 101
 2   2   5 = 225
```

Desimaalisen luvun muunto binäärimuotoon

Sääntö: Luku jaetaan 2:lla ja binääriluku muodostetaan jakojäännöksistä.

Esimerkki: Luvun 149 muunto binäärimuotoon.

```
2 |149      Jäännös  1
 2 |74        „      0
  2 |37       „      1
   2 |18      „      0
    2 |9      „      1 = 010 010 101
     2 |4     „      0
      2 |2    „      0
       2 |1   „      1
          0          ↑
```

Binääriluvun muunto desimaalimuotoon

Sääntö: Binääriluvun ensimmäinen numero kerrotaan 2:lla, tuloon lisätään seuraava numero, summa kerrotaan 2:lla jne. samoin kuin oktaalilukujen muunnossa.

```
10 010 101
 × 2
   2
 + 0 ←
   2
 × 2
   4
 + 0 ←
   4
 × 2
   8
 + 1 ←
   9
 × 2
  18
 + 0 ←
  18
 × 2
  36
 + 1 ←
  37
 × 2
  74
 + 0 ←
  74
 × 2
 148
 + 1 ←
 149
```

Tai 10 010 101

$= 1\,(2^7) + 0\,(2^6) + 0(2^5) + 1\,(2^4) +$

$0\,(2^3) + 1\,(2^2) + 0\,(2^1) + 1\,(2^0)$

$= 128 + 16 + 4 + 1$

$= 149$

MURTOLUKUJEN MUUNTO

Desimaalisen luvun muunto oktaalimuotoon

Sääntö: Lukua kerrotaan 8:lla ja muodostetaan oktaaliluku muistinumeroista.

Esimerkki: Luvun 0,149 muunto.

```
Lue    0,149
     x     8
   1 0,192
     x     8
   1 0,536
     x     8
   4 0,288
     x     8
   2 0,304
   = 0,1142 +
```

Oktaaliluvun muunto desimaalimuotoon

Sääntö: Luku 'hajoitetaan' 8:n potensseihin ja lasketaan yhteen.

Esimerkki:

$$
\begin{aligned}
01142 &= 1(8^{-1}) + 4(8^{-3}) + 2(8^{-4}) \\
&= 1/8 + 1/64 + 4/512 + 2/4096 \\
&= 610/4096 \\
&= 0{,}1489 \text{ tai } 0{,}149
\end{aligned}
$$

Oktaaliluvun muunto binäärimuotoon ja binääriluvun muunto oktaalimuotoon

Sääntö: Kokonaislukujen yhteydessä esitetty sääntö soveltuu myös murtolukuihin.

0,1	1	4	2	0,001	001	100 010
0,001	001	100	010	0,1	1	4 2

Binääriluvun muunto desimaalimuotoon

Sääntö: Sama kuin kokonaisluvuilla

Esimerkki:

$$
\begin{aligned}
0{,}001\ 001\ 100\ 010 &= 1(2^{-3}) + 1(2^{-6}) + 1(2^{-7}) + (2^{-11}) \\
&= 1/8 + 1/64 + 1/128 + 1/2048 \\
&= 305/2048 \\
&= 0{,}1489 \text{ plus} \\
&\text{tai } 0{,}149
\end{aligned}
$$

HEKSADESIMAALIJÄRJESTELMÄ

Koska kaikissa IBM:n tietokoneissa käytetään binäärijärjestelmää 2:n potenssien esittämiseen, perusmuotona Systeemin/360 ja ihmisen välisessä kommunikaatiossa on heksadesimaalijärjestelmä. Esim. käännösohjelmat kirjoittavat tavallisesti muistin sisällön heksadesimaalimuotoisena. Samoin koneen toimintaperiaatteita esittelevissä käsikirjoissa esim. käskyjen (operaatiokoodien) yhteydessä käytetään heksadesimaalista merkintätapaa.

Heksadesimaali tarkoittaa lukua 16. Heksadesimaalijärjestelmä on lukua 16 kantana käyttävä lukujärjestelmä, joka sisältää numerot 0-9 sekä lukuja 1-15 edustavat kirjaimet A-F. Heksadesimaalinumero esitetään neljällä bitillä (kuva 30).

DECIMAL SYSTEM	HEXADECIMAL SYSTEM	BINARY SYSTEM 8 4 2 1 Bit values
0	0	0 0 0 0
1	1	0 0 0 1
2	2	0 0 1 0
3	3	0 0 1 1
4	4	0 1 0 0
5	5	0 1 0 1
6	6	0 1 1 0
7	7	0 1 1 1
8	8	1 0 0 0
9	9	1 0 0 1
10	A	1 0 1 0
11	B	1 0 1 1
12	C	1 1 0 0
13	D	1 1 0 1
14	E	1 1 1 0
15	F	1 1 1 1

Kuva 30. Desimaali-, heksadesimaali- ja binäärijärjestelmien väliset suhteet

Koska Systeemin/360 tavussa on 8 bittiä (ja pariteettibitti) jokainen tavu sisältää siten kaksi heksadesimaalinumeroa, esimerkiksi:

desimaalinen luku		248	voidaan esittää
binäärisenä	1111	1000	ja edelleen
heksadesimaalisena	F	8	

On muistettava, että heksadesimaalijärjestelmä on vain eräs binääriesitykseen verrattuna lyhyempi esitysmuoto kuten oktaalijärjestelmäkin. Siten se soveltuu aikaisemmin esitettyihin koodijärjestelmiin. EBCDIC-koodissa edellisen esimerkin F8 merkitsee lukua 8. USASCII-8 -koodissa se merkitsee kirjainta x jne. Täysin koodista riippumatta kyseinen merkki voidaan ilmaista heksadesimaalilukuna F8.

Kokonaislukujen muunto heksadesimaalijärjestelmästä desimaalijärjestelmään

Esimerkiksi voidaan ottaa vaikkapa heksadesimaaliluku A4B5. Ensinnäkin voidaan heksadesimaalinumeroille antaa järjestysluvut siten, että oikeanpuoleisimmalle (eli alimmalle) positiolle annetaan numero 1 ja siitä edelleen vasemmalle 2, 3, 4 jne. Kuvassa 31 olevan taulukon mukaan saadaan

5 positiossa	1	=	5
B positiossa	2	=	176
4 positiossa	3	=	1024
A positiossa	4	=	40960
Luvun A4B5 arvo on summa			42165

Huom. Ellei taulukoita ole käytettävissä, voidaan käyttää samaa menetelmää kuin oktaalilukuja muunnettaessa vaihtamalla 8:n tilalle kertojaksi luku 16.

Kokonaislukujen muunto desimaalijärjestelmästä heksadesimaalijärjestelmään

Esim. luvun 16.428 muuntamiseksi heksadesimaalimuotoon edellistä menetelmää voidaan käyttää käännettynä, toisin sanoen etsitään kuvan 31 taulukosta lukua 16.428 lähinnä pienempi luku eli 16.384. Tämä luku vähennetään alkuperäisestä eli 16.428:sta ja saatu erotus käsitellään kuten edellä. Menettely toistetaan kunnes erotus jää pienemmäksi kuin 16. Koko proseduuri on siis seuraavanlainen:

Desimaaliluku, jolle etsitään heksadesimaalinen vastine
(eli muunnettava luku)

Heksadesimaalinumero 4, positio 4		16.428
	=	16.384
Jäännös		44
Numero 0, positio 3	=	0
Jäännös		44
Numero 2, positio 2	=	32
Jäännös		12
Numero C, positio 1	=	12

Luvun 16.428 heksadesimaalinen vastine on siis 402C.

Huom. Ellei taulukkoa ole käytettävissä, voidaan käyttää samaa menetelmää kuin desimaalilukuja oktaalimuotoon muunnettaessa korvaamalla jakajana oleva 8 luvulla 16.

Murtolukujen muunto heksadesimaalijärjestelmästä desimaalijärjestelmään

Tässä tapauksessa käytetään kuvassa 32 olevaa taulukkoa, joka sisältää mantissan kunkin position desimaaliset arvot
Esimerkki: Luvun ,ABC muunto desimaaliseen muotoon

,A, positio 1	0,6250
,0B, positio 2	0,0429 6875
,00C, positio 3	0,0029 2968 7500

Haettu luku on edellisten summa

0,6708 9843 7500

HUOM. Ilman taulukoita muunto suoritetaan samoin kuin oktaaliluvuilla edellyttäen tietenkin, että kantaluku 8 korvataan 16:lla.

Murtolukujen muunto desimaalijärjestelmästä heksadesimaalijärjestelmään

Taulukosta 32 etsitään muunnettavaa lukua lähinnä pienempi luku ja vähennetään se alkuperäisestä luvusta. Menettely toistetaan saadun erotuksen suhteen niin monta kertaa kuin desimaaleja halutaan.
Esimerkki: Luvun ,13 muunto heksadesimaalimuotoon.

		Heksadesimaalinen vastine
Muunnettava desimaaliluku	0,1300	
Lähinnä pienempi des. luku	0,1250	= 0,2
Erotus	0,0050 0000	
Lähinnä pienempi des. luku	0,0039 0625	= 0,01
Erotus	0,0010 9375 0000	
Lähinnä pienempi des. luku	0,0009 7656 2500	= 0,004
Erotus	0,0001 1718 7500	
Lähinnä pienempi des. luku	0,0001 0681 1523 4375	= 0,0007
Luvun 0,13 heksadesimaalinen vastine on edellisten summa		= 0,2147

Huom. Ilman taulukoita muunto suoritetaan samoin kuin muunto oktaalijärjestelmään edellyttäen, että kantaluku 8 korvataan 16:lla.

HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	268,435,456	1	16,777,216	1	1,048,576	1	65,536	1	4,096	1	256	1	16	1	1
2	536,870,912	2	33,554,432	2	2,097,152	2	131,072	2	8,192	2	512	2	32	2	2
3	805,306,368	3	50,331,648	3	3,145,728	3	196,608	3	12,288	3	768	3	48	3	3
4	1,073,741,824	4	67,108,864	4	4,194,304	4	262,144	4	16,384	4	1,024	4	64	4	4
5	1,342,177,280	5	83,886,080	5	5,242,880	5	327,680	5	20,480	5	1,280	5	80	5	5
6	1,610,612,736	6	100,663,296	6	6,291,456	6	393,216	6	24,576	6	1,536	6	96	6	6
7	1,879,048,192	7	117,440,512	7	7,340,032	7	458,752	7	28,672	7	1,792	7	112	7	7
8	2,147,483,648	8	134,217,728	8	8,388,608	8	524,288	8	32,768	8	2,048	8	128	8	8
9	2,415,919,104	9	150,994,944	9	9,437,184	9	589,824	9	36,864	9	2,304	9	144	9	9
A	2,684,354,560	A	167,772,160	A	10,485,760	A	655,360	A	40,960	A	2,560	A	160	A	10
B	2,952,790,016	B	184,549,376	B	11,534,336	B	720,896	B	45,056	B	2,816	B	176	B	11
C	3,221,225,472	C	201,326,592	C	12,582,912	C	786,432	C	49,152	C	3,072	C	192	C	12
D	3,489,660,928	D	218,103,808	D	13,631,488	D	851,968	D	53,248	D	3,328	D	208	D	13
E	3,758,096,384	E	234,881,024	E	14,680,064	E	917,504	E	57,344	E	3,584	E	224	E	14
F	4,026,531,840	F	251,658,240	F	15,728,640	F	983,040	F	61,440	F	3,840	F	240	F	15
Hexadecimal Positions	8		7		6		5		4		3		2		1

Kuva 31. Heksadesimaalilukujen muuntotaulukko (kokonaisluvut)

HEX	0123 DEC	HEX	4 5 6 7 DECIMAL	HEX	0 1 2 3 DECIMAL	HEX	4 5 6 7 DECIMAL EQUIVALENT
.0	.0000	.00	.0000 0000	.000	.0000 0000 0000	.0000	.0000 0000 0000 0000
.1	.0625	.01	.0039 0625	.001	.0002 4414 0625	.0001	.0000 1525 8789 0625
.2	.1250	.02	.0078 1250	.002	.0004 8828 1250	.0002	.0000 3051 7578 1250
.3	.1875	.03	.0117 1875	.003	.0007 3242 1875	.0003	.0000 4577 6367 1875
.4	.2500	.04	.0156 2500	.004	.0009 7656 2500	.0004	.0000 6103 5156 2500
.5	.3125	.05	.0195 3125	.005	.0012 2070 3125	.0005	.0000 7629 3945 3125
.6	.3750	.06	.0234 3750	.006	.0014 6484 3750	.0006	.0000 9155 2734 3750
.7	.4375	.07	.0273 4375	.007	.0017 0898 4375	.0007	.0001 0681 1523 4375
.8	.5000	.08	.0312 5000	.008	.0019 5312 5000	.0008	.0001 2207 0312 5000
.9	.5625	.09	.0351 5625	.009	.0021 9726 5625	.0009	.0001 3732 9101 5625
.A	.6250	.0A	.0390 6250	.00A	.0024 4140 6250	.000A	.0001 5258 7890 6250
.B	.6875	.0B	.0429 6875	.00B	.0026 8554 6875	.000B	.0001 6784 6679 6875
.C	.7500	.0C	.0468 7500	.00C	.0029 2968 7500	.000C	.0001 8310 5468 7500
.D	.8125	.0D	.0507 8125	.00D	.0031 7382 8125	.000D	.0001 9836 4257 8125
.E	.8750	.0E	.0546 8750	.00E	.0034 1796 8750	.000E	.0002 1362 3046 8750
.F	.9375	.0F	.0585 9375	.00F	.0036 6210 9375	.000F	.0002 2888 1835 9375
Hexadecimal Positions	1		2		3		4

Kuva 32. Heksadesimaalilukujen muuntotaulukko (murto-osat)

TIETOJEN TALLENNUSVÄLINEET

Reikäkortti

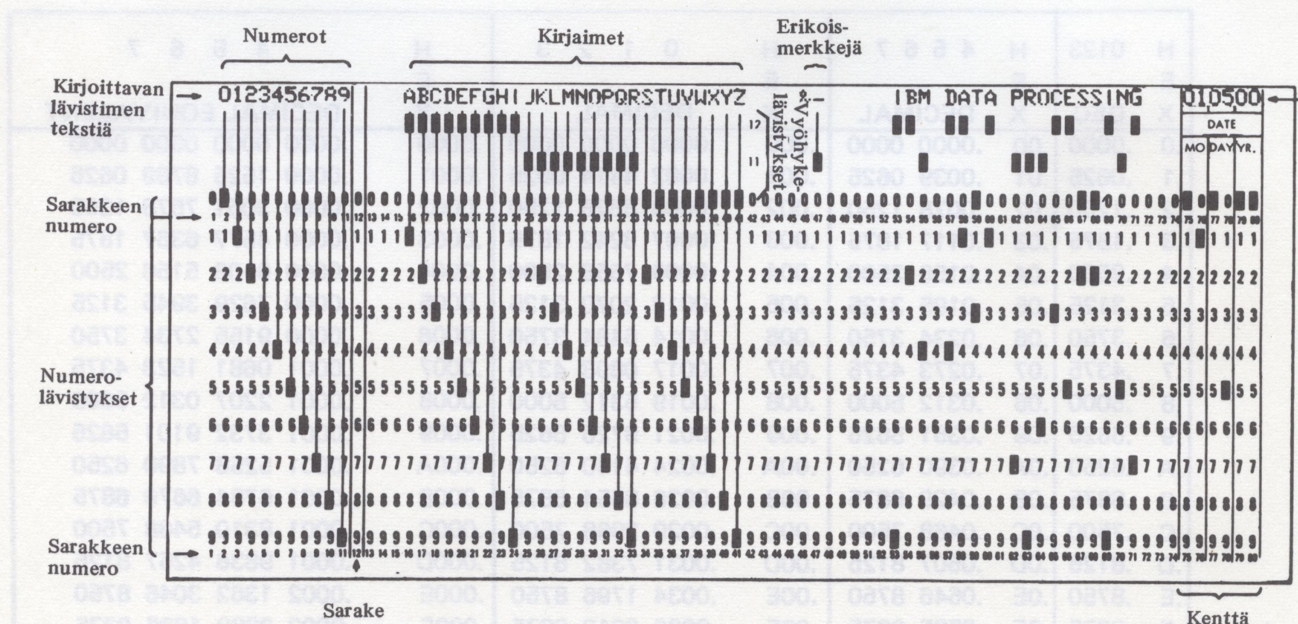
Eräs 'suosituimmista' koneiden ja ihmisen väliseen kommunikatioon käytettävistä välineistä on reikäkortti. Tiedot tallennetaan pieninä nelikulmaisina reikinä, jotka lävistetään määrättyihin paikkoihin standardikokoiseen korttiin (kuva 33). Tiedot, joita määrätyissä paikoissa esiintyvät tai niistä puuttuvat reiät esittävät, voidaan lukea kortin liikkuessa kortinlukulaitteen läpi.

Korttien lukeminen eli tunnistaminen on toiminta, jossa reikinä tallennetut tiedot automaattisesti muutetaan sähköiseen muotoon ja sijoitetaan koneeseen. Kortteja käytetään sekä tietojen syöttämiseen että tallennettavan tiedon tulostamiseen lävistämällä tiedot korteille. Siten reikäkorttien käyttö ei rajoitu ainoastaan tietojen siirtämiseen muualta koneeseen, vaan niitä voidaan käyttää myös siirtämään tietoja koneesta toiseen.

Reikäkorteissa on 80 pystysuoraa saraketta, kussakin 12 lävistettävää kohtaa. Nämä 12 kohtaa muodostavat 12 vaakasuoraa riviä kortin poikki. Yksi tai useampia reikiä sarakkeessa esittää merkkiä. Käytettyjen sarakkeiden lukumäärä riippuu esitettävän tiedon laajuudesta.

Korttia nimitetään usein tietojaksoyksiköksi, koska tieto rajoittuu 80 sarakkeeseen ja koska kortti luetaan ja lävistetään tietoyksikkönä. Kuitenkin kortissa oleva tieto voi käsittää osan tietojaksosta, yhden tietojakson tai useita jaksoja. Jos tarvitaan enemmän kuin 80 saraketta tietojakson esittämistä varten, voidaan käyttää kahta tai useampaa korttia. Tietojakson jatkuminen kortista toiseen voidaan saada aikaan lävistämällä identifioimismerkki kussakin kortissa olevaan määrättyyn sarakkeeseen.

Korttiin lävistetty tieto luetaan eli tulkitaan laitteella, jota sanotaan kortinlukijaksi, ja korttiin lävistetään tietoa laitteella, jota sanotaan korttilävistimeksi. Tiedot siirretään perustositteista reikäkorteille käsinohjattavilla lävistyskoneilla.



Kuva 33. Reikäkortin vakiokoodi

Sana 1 Sana 54

Kortille voidaan sijoittaa 54 sanaa (1-1/3 saraketta per sana eli kolmea sanaa varten tarvitaan neljä saraketta). Sanojen järjestysnumerot ovat kortin yläosassa joka kolmannen sarakkeen kohdalla.

Kuva 34. IBM 1130 binäärikortti

Reikäkorttikoodi

Normaalireikäkorttikoodi käyttää kahtatoista mahdollista lävistyskohtaa kortin pystysuorassa sarakkeessa esittämään numeroita, kirjaimia ja erikoismerkkejä (kuva 33). Nämä kaksitoista lävistystä jaetaan kahteen ryhmään: numero- ja vyöhykelävistyksiin. Yhdeksän ensimmäistä lävistystä alhaalta lukien ovat numerolävistyksiä ja niillä on arvot 9, 8, 7, 6, 5, 4, 3, 2 ja 1. Loput kolme lävistystä, 0, 11 ja 12 ovat vyöhykelävistyksiä. 0-lävistystä käytetään sekä numero- että vyöhykelävistyksenä.

Numerot 0-9 esitetään pystysuorassa sarakkeessa olevalla yhdellä reiällä. Esimerkiksi 0 esitetään yhdellä reiällä, joka lävistetään sarakkeen 0-kohtaan.

Kirjaimet esitetään kahdella samassa pystysuorassa sarakkeessa olevalla reiällä. Kirjaimissa A-I käytetään 12-lävistystä ja numerolävistyksiä 1-9. Kirjaimissa J-R käytetään 11-lävistystä ja numerolävistyksiä 1-9. Kirjainten S-Z esittämiseen käytetään vyöhykelävistyksenä 0-lävistystä ja numerolävistyksenä on jokin lävistyksistä 2-9.

Tavalliset erikoismerkit ⋆ $ % jne. esitetään samassa sarakkeessa olevilla yhdellä, kahdella tai kolmella lävistyksellä käyttäen sellaisia yhdistelmiä, joita ei ole varattu numeroita tai kirjaimia varten.

Binäärisarakkeinen tietojen tallennus

Binäärisarakkeisella tarkoitetaan menetelmää, jolla tiedot tallennetaan reikäkortille binäärimuodossa. Tässä menetelmässä koko reikäkortin voidaan ajatella olevan yhtenäinen bittijono, joka alkaa ensimmäisen sarakkeen riviltä 12, kulkee saman sarakkeen rivin 9 kautta seuraavaan sarakkeeseen ja niin edespäin sarakkeen 80 riville 9 asti. Tieto esitetään sanoina. Jokainen lävistys edustaa sanan kyseisessä positiossa olevaa binääriykköstä. Tietokonejärjestelmä IBM 1130 sisältää lisälaitteen, joka tarvitaan tällaisten binäärikorttien käsittelyä varten. Tässä järjestelmässä 16 bittiä muodostaa sanan, jonka esittämiseen tarvitaan siten 1 1/3 saraketta. Kuvassa 34 on esitetty IBM 1130-järjestelmän binäärikortti.

Reikänauha

Lävistettyä paperinauhaa käytetään suunnilleen samaan tarkoitukseen kuin reikäkortteja. Se on alunperin kehitetty lennätintä varten, mutta nykyään sitä käytetään myös muissa koneissa. Siirrettäessä tietoa pitkien matkojen päähän kone siirtää tiedon reikäkorteilta reikänauhalle ja lähettää tiedon puhelin- tai lennätinlinjaa pitkin. Vastaanottopaikalla lävistetään samanlainen paperinauha, josta tiedot siirretään reikäkorteille myöhempää käsittelyä varten.

Tiedot tallennetaan erikoisina lävistettyjen pyöreiden reikien yhdistelminä, jotka on tarkasti sijoitettu paperinauhalle (kuvat 35 ja 36). Reikänauhalle voidaan tallentaa tietoa jatkuvasti, kun taas reikäkortit ovat määrätyn pituisia. Reikänauhaa voidaan siten käyttää tallentamaan tietojaksoja, joiden pituutta rajoittavat vain sen muistin suuruus, johon tieto sijoitetaan tai josta se otetaan.

Reikänauhalla olevat tiedot luetaan nauhanlukijalla ja lävistäminen suoritetaan nauhanlävistimellä.

Kahdeksankanavainen koodi

Tieto tallennetaan ja luetaan käyttäen reikiä, jotka sijoitetaan paperinauhalle kahdeksaan yhdensuuntaiseen riviin eli kanavaan. Yhtä saraketta, jossa on kahdeksan lävistettävää kohtaa (yksi kutakin riviä kohti), käytetään koodattaessa numeroita, kirjaimia, erikoismerkkejä ja toimintamerkkejä. Kuvassa 35 esitetään osa 8-kanavaisesta paperinauhasta, johon on lävistetty muutamia koodattuja merkkejä.

Nauhan neljälle alimmalle riville (nauhan syöttöä varten olevia reikiä ei oteta huomioon) on annettu arvot 1, 2, 4 ja 8 ja niillä esitetään numerot 0-9. Arvojen summa vastaa kyseessä olevan numeron arvoa. Esimerkiksi reikä 1-rivillä esittää numeroa 1. 1- ja 2-rivillä olevien reikien yhdistelmä esittää numeroa 3.

X- ja 0-rivit vastaavat reikäkortien vyöhykelävistyksiä. Näiden ja numerorivien yhdistelmiä käytetään esittämään kirjaimia ja erikoismerkkejä. Kirjaimien ja erikoismerkkien koodaus esitetään kuvassa 35.

Jotta voitaisiin varmistua jokaisen merkin oikeasta lävistyksestä lävistetään kuhunkin sarakkeeseen pariton määrä reikiä. Tarkistusreikä täytyy olla jokaisessa sellaisessa sarakkeessa, jonka peruskoodi (X, 0, 8, 4, 2, 1) käsittää parillisen määrän reikiä.

EL-sarakkeessa oleva reikä on erikoismerkki, joka osoittaa tietojakson päättymisen nauhalla (END OF LINE). Nauhansyöttökoodi (TAPE FEED) käsittää lävistykset riveissä X, 0, 8, 4, 2 ja 1. Sitä käytetään merkitsemään tyhjää. Nauhanlukija hyppää automaattisesti syöttölävistyksen yli.

Viisikanavainen koodi

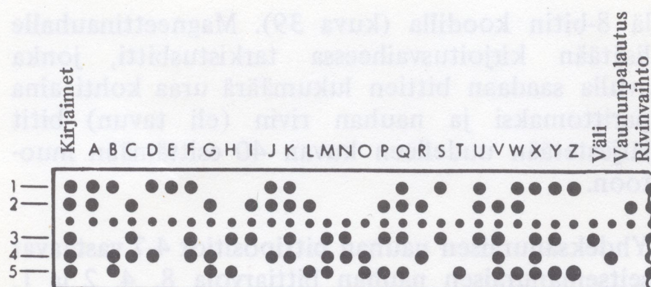
Tieto tallennetaan ja luetaan käyttäen reikiä, jotka sijoitetaan nauhalle viiteen samansuuntaiseen kanavaan eli riviin. Yhtä viisi lävistettävää kohtaa käsittävää saraketta käytetään jokaisen numeron, kirjaimen, erikois- tai toimintamerkin koodaukseen. Kuvassa 36 esitetään osa viisikanavaisesta nauhasta, johon on lävistetty muutamia koodattuja merkkejä.

Koska viidellä lävistyskohdalla voidaan saada vain 32 erilaista reikäyhdistelmää, kaksinkertaistetaan koodimerkkien lukumäärä vaihtojärjestelmällä. Kun merkkejä edeltää kirjainkoodi (LTRS), ne tulkitaan kirjaimiksi. Kun merkkejä edeltää numerokoodi (FIGS), merkit tulkitaan numeroiksi tai erikoismerkeiksi (kuva 36).

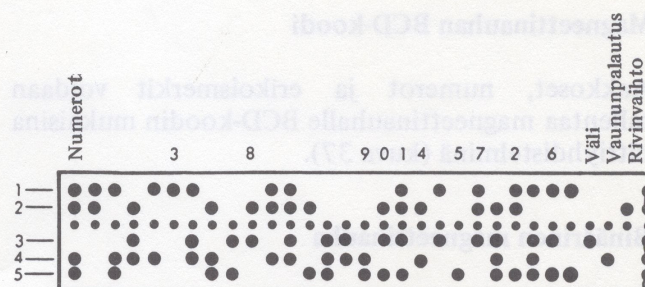
Kymmentä näistä 32:sta koodista käytetään esittämään sekä kirjaimia P, Q, W, E, R, T, Y, U, I ja O että vastaavasti numeroita 0-9. Merkkien tulkinta riippuu niitä edeltäneestä vaihtokoodista. Samoin erikoismerkkikoodit ovat samanlaisia kuin muut kirjaimet. Koneen käyttäjä voi määrätä, mitkä erikoismerkit ja kirjaimet vastaavat toisiaan.

Toimintamerkit, joita ovat välike, (SPACE), vaunun palautus (CR) ja rivin siirto (LF), ovat

Kuva 35. Reikänauha, kahdeksan kanavan koodi



Kuva 36. Reikänauha, viiden kanavan koodi



samoja sekä LTRS- että FIGS-järjestelmässä. Välikettä käytetään osoittamaan ettei sillä kohdalla nauhaa ole tietoa. CR- ja LF-merkkien kulloinenkin tarkoitus riippuu käytetystä koneesta.

Magneettinauha

Magneettinauha on eräs tietokonejärjestelmien syöttö- ja tulostustietojen tallennuksen perusvälineistä. Sitä käytetään paljon myös välitulosten tallentamiseen ja suurten tiedostojen muodostamiseen.

Magneettinauhan avulla saavutetaan tietoja koneeseen syötettäessä suuri nopeus ja samoin tapahtuu käsitellyn tiedon tallennus tehokkaasti ja erittäin nopeasti. On mahdollista siirtää aina 640.000 numeerista merkkiä sekunnissa ja lisäksi täysin luotettavasti.

Magneettinauhayksikkö toimii tietokonejärjestelmissä sekä syöttö- että tulostusyksikkönä. Se siirtää nauhan luku-kirjoituspään läpi ja suorittaa varsinaisen luku-kirjoitustoiminnan.

Tieto tallennetaan magneettinauhalle magneettisina pisteinä, joita sanotaan biteiksi.

Tieto voidaan säilyttää muuttumattomana kuinka kauan tahansa tai se voidaan automaattisesti hävittää, ja nauha voidaan käyttää uudestaan useita kertoja luotettavuuden vähentymättä.

Jotta nauhaa voitaisiin helposti käsitellä, se kelataan kelalle ja pölyttömään nauhakoteloon. Erillisillä keloilla oleva nauha on 1/2 tuumaa leveä ja kelalla voi olla nauhaa 2400 jalan pituudelta.

IBM 2420 nauhayksikön (malli 7) rakenne helpottaa tuntuvasti nauhan asetusta - nauhan pujotus kasetista vastaanottokelalle, siirto tyhjiökanaviin sekä nauhan alun haku tapahtuvat täysin automaattisesti. Täyden kelan (2400 jalkaa) takaisinkelaus kestää vain minuutin, koska nauhaa ei tarvitse poistaa tyhjiökanavista.

Tieto tallennetaan nauhalle samansuuntaisiin uriin eli kanaviin. Nauhalla olevat urat muodostavat yhden tietosarakkeen. Pystysuorien sarakkeiden välinen välike muodostetaan automaattisesti nauhalle kirjoitettaessa ja sen pituus vaihtelee tallennettaessa käytettävän merkkitiheyden mukaan. 1/2 tuuman nauhalle voidaan tallentaa jopa 1600 merkkiä tuumalle.

Nauhalla olevien tietojaksojen väliin muodostetaan automaattisesti pitempi välike. Tätä välikettä sanotaan lohkoväliksi.

Seitsemänuraisilla magneettinauhoilla voidaan käyttää kahta eri koodia, binääristä tai BCD-koodia. Koodi riippuu käytettävästä tietokoneesta.

IBM 2400 sarjan nauhayksiköissä on yleensä 9-urainen luku-kirjoituspää, mutta siihen on saatavissa myös 7-uran luku-kirjoituspää. Normaali luku-kirjoituspää, siis 9-urainen, pystyy lukemaan mallista riippuen joko 800 tai 1600 tavua per tuuma 9 uralle tallennettua tietoa. Sama pätee tietysti myös nauhalle kirjoittamiseen. Tavu sisältää 8 tietobittiä sekä tarkistusbitin. Yhdellä tavulla voidaan esittää 2 pakattua numeroa, 8 binääribittiä tai yksi kirjain tai erikoismerkki.

Pakatun muodon käyttö koskee vain numeroita, mutta tietyissä tapauksissa tästä saattaa olla huomattavaakin hyötyä, koska yhteen nauhan 'riviin' voidaan tallentaa kaksi numeroa ja siten yksinkertaistaa numeeristen tietojen luku- ja kirjoitusnopeus.

Magneettinauhan BCD-koodi

Aakkoset, numerot ja erikoismerkit voidaan tallentaa magneettinauhalle BCD-koodin mukaisina bittiyhdistelminä (kuva 37).

Binäärinen magneettinauha

Joissakin tietokonejärjestelmissä tiedot tallennetaan magneettinauhalle binäärimuodossa.

Kuten BCD-koodissa, käytetään C-uraa tässäkin tapauksessa nauhan luvun ja kirjoituksen oikeellisuuden tarkistamiseen. Binäärimuotoa käytettäessä on jokaisella rivillä oltava pariton määrä ykkösbittejä.

Nauhan pituussuunnassa tapahtuva tarkistus on sama kuin BCD-koodia käytettäessä. Kultakin uralta kerätyn bittien lukumäärän per lohko on oltava parillinen.

Yhdeksänurainen magneettinauha

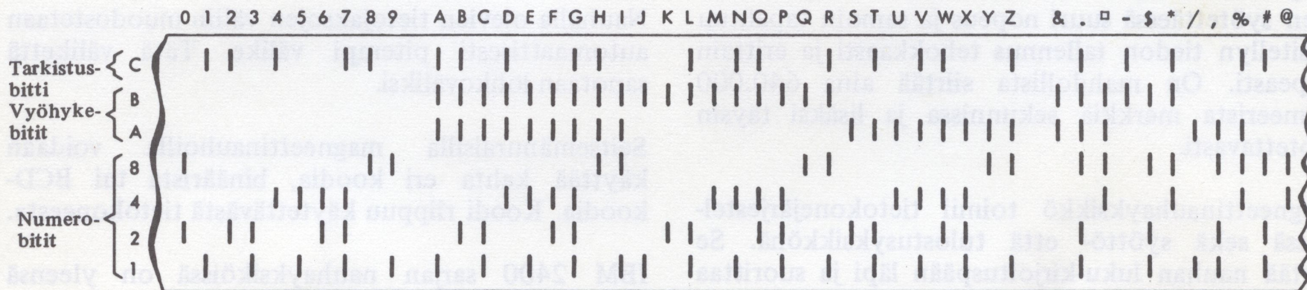
IBM 2400 sarjan yhdeksänuraisille nauhoille voidaan tallentaa tietoja Systeemin/360 käyttämällä 8-bitin koodilla (kuva 39). Magneettinauhalle lisätään kirjoitusvaiheessa tarkistusbitti, jonka avulla saadaan bittien lukumäärä uraa kohti aina parittomaksi ja nauhan rivin (eli tavun) bitit järjestetään uudelleen kuvan 40 esittämään muotoon.

Yhdeksänuraisen nauhan bittipositiot 4-7 vastaavat seitsemänuraisen nauhan bittiarvoja 8, 4, 2 ja 1. Positioiden 2 ja 3 merkitys on päinvastainen kuin seitsemänuraisen nauhan A- ja B-positioiden. Ylimääräisten bittipositioiden 0 ja 1 avulla merkit voidaan jakaa neljään ryhmään: isot kirjaimet ja numerot, pienet kirjaimet, erikoismerkit, sekä merkit, bittiyhdistelmät joilla ei ole graafista vastinetta eikä muuta 'tehtävää'. On huomattava, että nauhan urat eivät ole 0-7 järjestyksessä.

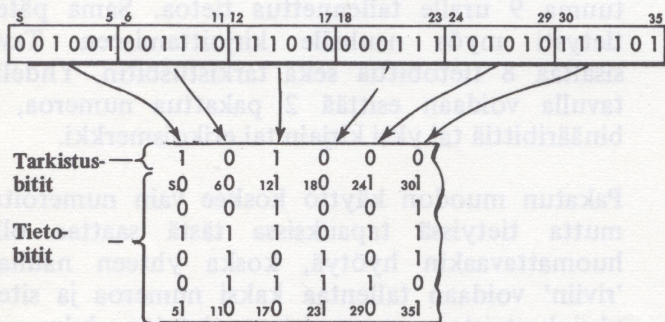
Lomakkeet

Magneettisesti luettavat merkit

Toinen paperia käyttävä käsiteltävien tietojen ilmaisumenetelmä on magneettisella musteella kirjoitetut merkit - ne muodostavat kielen, jota sekä ihminen että kone pystyvät lukemaan. Magneettiset merkit painetaan paperille kuten esitetään kuvassa 41. Merkkien muoto sallii niiden



Kuva 37. Magneettinauha, seitsemän kanavan, 7-bitin aakkosnumeerinen koodi



Kuva 38. Seitsemänkanavainen magneettinauha, binäärikoodi

visuaalisen tulkinnan. Erikoinen magneettinen muste tekee mahdolliseksi merkkien koneellisen tulkinnan.

Magneettisten merkkien painaminen paperille suoritetaan koneella. Tositteet voivat olla paperia tai kortteja, ja niiden koko voi vaihdella. Pituus täytyy kuitenkin olla 6 - 8 3/4 tuumaa, leveys 2 3/4 - 3 2/3 tuumaa ja paksuus 0.003 - 0.007 tuumaa.

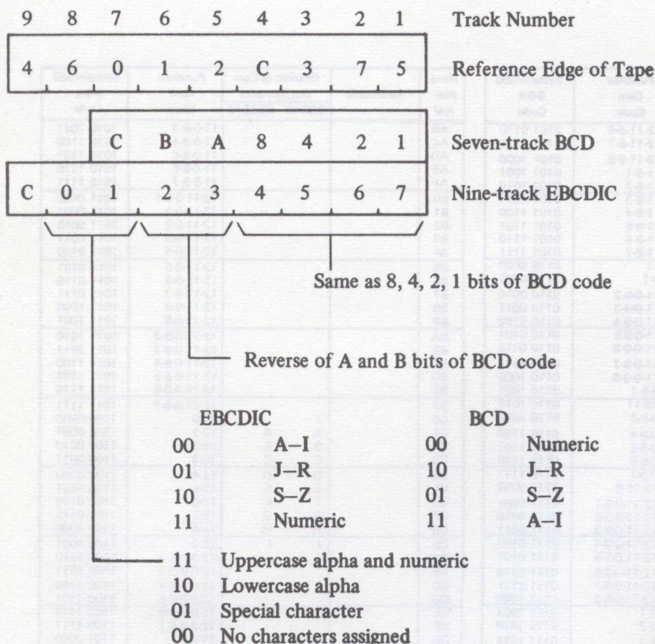
Lomakkeiden merkintää varten on käytettävissä erityinen laite, IBM 1260 merkintälaite, joka lisäksi voi suorittaa eräitä tarkistustoimintoja. Merkinnän jälkeen IBM 1419 magneettimerkkien lukija lukee tiedot suoraan lomakkeista ja muuntaa ne konekieliseen muotoon. Samalla tiedot tallentuvat keskusmuistiin. Luvun yhteydessä IBM 1419 lukija pystyy myös lajittelemaan lomakkeita.

Hexadecimal	Mnemonic	Graphic & Control Symbols BCDIC	EBCDIC	Punched Card Code	System/360 8-Bit Code
00			NUL	12-0-9-8-1	0000 0000
01				12-9-1	0000 0001
02				12-9-2	0000 0010
03				12-9-3	0000 0011
04	SPM		PF	12-9-4	0000 0100
05	BALR		HT	12-9-5	0000 0101
06	BCTR		LC	12-9-6	0000 0110
07	BCR		DEL	12-9-7	0000 0111
08	SSK			12-9-8	0000 1000
09	ISK			12-9-8-1	0000 1001
0A	SVC			12-9-8-2	0000 1010
0B				12-9-8-3	0000 1011
0C	(EBCDIC +)			12-9-8-4	0000 1100
0D	(EBCDIC –)			12-9-8-5	0000 1101
0E				12-9-8-6	0000 1110
0F				12-9-8-7	0000 1111
10	LPR			12-11-9-8-1	0001 0000
11	LNR			11-9-1	0001 0001
12	LTR			11-9-2	0001 0010
13	LCR			11-9-3	0001 0011
14	NR		RES	11-9-4	0001 0100
15	CLR		NL	11-9-5	0001 0101
16	OR		BS	11-9-6	0001 0110
17	XR		IL	11-9-7	0001 0111
18	LR			11-9-8	0001 1000
19	CR			11-9-8-1	0001 1001
1A	AR			11-9-8-2	0001 1010
1B	SR			11-9-8-3	0001 1011
1C	MR			11-9-8-4	0001 1100
1D	DR			11-9-8-5	0001 1101
1E	ALR			11-9-8-6	0001 1110
1F	SLR			11-9-8-7	0001 1111
20	LPDR			11-0-9-8-1	0010 0000
21	LNDR			0-9-1	0010 0001
22	LTDR			0-9-2	0010 0010
23	LCDR			0-9-3	0010 0011
24	HDR		BYP	0-9-4	0010 0100
25			LF	0-9-5	0010 0101
26			EOB	0-9-6	0010 0110
27			PRE	0-9-7	0010 0111
28	LDR			0-9-8	0010 1000
29	CDR			0-9-8-1	0010 1001
2A	N ADR		SM	0-9-8-2	0010 1010
2B	N SDR			0-9-8-3	0010 1011
2C	N MDR			0-9-8-4	0010 1100
2D	N DDR			0-9-8-5	0010 1101
2E	AWR			0-9-8-6	0010 1110
2F	SWR			0-9-8-7	0010 1111
30	LPER			12-11-0-9-8-1	0011 0000
31	LNER			9-1	0011 0001
32	LTER			9-2	0011 0010
33	LCER			9-3	0011 0011
34	HER		PN	9-4	0011 0100
35			RS	9-5	0011 0101
36			UC	9-6	0011 0110
37			EOT	9-7	0011 0111
38	LER			9-8	0011 1000
39	CER			9-8-1	0011 1001
3A	N AER			9-8-2	0011 1010
3B	N SER			9-8-3	0011 1011
3C	N MER			9-8-4	0011 1100
3D	N DER			9-8-5	0011 1101
3E	AUR			9-8-6	0011 1110
3F	SUR			9-8-7	0011 1111
40	STH		SP	no punches	0100 0000
41	LA			12-0-9-1	0100 0001
42	STC			12-0-9-2	0100 0010
43	IC			12-0-9-3	0100 0011
44	EX			12-0-9-4	0100 0100
45	BAL			12-0-9-5	0100 0101
46	BCT			12-0-9-6	0100 0110
47	BC			12-0-9-7	0100 0111
48	LH			12-0-9-8	0100 1000
49	CH			12-8-1	0100 1001
4A	AH		¢	12-8-2	0100 1010
4B	SH	.	.	12-8-3	0100 1011
4C	MH	⌑)	<	12-8-4	0100 1100
4D		[	(	12-8-5	0100 1101
4E	CVD	<	+	12-8-6	0100 1110
4F	CVB	‡	|	12-8-7	0100 1111
50	ST	& +	&	12	0101 0000
51				12-11-9-1	0101 0001
52				12-11-9-2	0101 0010
53				12-11-9-3	0101 0011
54	N			12-11-9-4	0101 0100
55	CL			12-11-9-5	0101 0101
56	O			12-11-9-6	0101 0110
57	X			12-11-9-7	0101 0111
58	L			12-11-9-8	0101 1000
59	C			11-8-1	0101 1001
5A	A		!	11-8-2	0101 1010
5B	S	$	$	11-8-3	0101 1011
5C	M	*	*	11-8-4	0101 1100
5D	D	]	)	11-8-5	0101 1101
5E	AL	;	;	11-8-6	0101 1110
5F	SL	Δ	¬	11-8-7	0101 1111
60	STD	-	-	11	0110 0000
61		/	/	0-1	0110 0001
62				11-0-9-2	0110 0010
63				11-0-9-3	0110 0011
64				11-0-9-4	0110 0100
65				11-0-9-5	0110 0101
66				11-0-9-6	0110 0110
67				11-0-9-7	0110 0111
68	LD			11-0-9-8	0110 1000
69	CD			0-8-1	0110 1001
6A	N AD			12-11	0110 1010
6B	N SD	,	,	0-8-3	0110 1011
6C	N MD	% (	%	0-8-4	0110 1100
6D	N DD	⋎	_	0-8-5	0110 1101
6E	AW	\	>	0-8-6	0110 1110
6F	SW	⧻	?	0-8-7	0110 1111
70	STE			12-11-0	0111 0000
71				12-11-0-9-1	0111 0001
72				12-11-0-9-2	0111 0010
73				12-11-0-9-3	0111 0011
74				12-11-0-9-4	0111 0100
75				12-11-0-9-5	0111 0101
76				12-11-0-9-6	0111 0110
77				12-11-0-9-7	0111 0111
78	LE			12-11-0-9-8	0111 1000
79	CE			8-1	0111 1001
7A	N AE	b̸	:	8-2	0111 1010
7B	N SE	# =	#	8-3	0111 1011
7C	N ME	@'	@	8-4	0111 1100
7D	N DE	:	'	8-5	0111 1101
7E	AU	>	=	8-6	0111 1110
7F	SU	✓	"	8-7	0111 1111
80	SSM			12-0-8-1	1000 0000
81			a	12-0-1	1000 0001
82	LPSW		b	12-0-2	1000 0010
83	(Diagnose)		c	12-0-3	1000 0011
84	WRD		d	12-0-4	1000 0100
85	RDD		e	12-0-5	1000 0101
86	BXH		f	12-0-6	1000 0110
87	BXLE		g	12-0-7	1000 0111
88	SRL		h	12-0-8	1000 1000
89	SLL		i	12-0-9	1001 1001
8A	SRA			12-0-8-2	1000 1010
8B	SLA			12-0-8-3	1000 1011
8C	SRDL			12-0-8-4	1000 1100
8D	SLDL			12-0-8-5	1000 1101
8E	SRDA			12-0-8-6	1000 1110
8F	SLDA			12-0-8-7	1000 1111
90	STM			12-11-8-1	1001 0000
91	TM		j	12-11-1	1001 0001
92	MVI		k	12-11-2	1001 0010
93	TS		l	12-11-3	1001 0011
94	NI		m	12-11-4	1001 0100
95	CLI		n	12-11-5	1001 0101
96	OI		o	12-11-6	1001 0110
97	XI		p	12-11-7	1001 0111
98	LM		q	12-11-8	1001 1000
99			r	12-11-9	1001 1001
9A				12-11-8-2	1001 1010
9B				12-11-8-3	1001 1011
9C	SIO			12-11-8-4	1001 1100
9D	TIO			12-11-8-5	1001 1101
9E	HIO			12-11-8-6	1001 1110
9F	TCH			12-11-8-7	1001 1111
A0				11-0-8-1	1010 0000
A1				11-0-1	1010 0001
A2			s	11-0-2	1010 0010
A3			t	11-0-3	1010 0011
A4			u	11-0-4	1010 0100
A5			v	11-0-5	1010 0101
A6			w	11-0-6	1010 0110
A7			x	11-0-7	1010 0111
A8			y	11-0-8	1010 1000
A9			z	11-0-9	1010 1001
AA				11-0-8-2	1010 1010
AB				11-0-8-3	1010 1011
AC				11-0-8-4	1010 1100
AD				11-0-8-5	1010 1101
AE				11-0-8-6	1010 1110
AF				11-0-8-7	1010 1111
B0				12-11-0-8-1	1011 0000
B1				12-11-0-1	1011 0001
B2				12-11-0-2	1011 0010
B3				12-11-0-3	1011 0011
B4				12-11-0-4	1011 0100
B5				12-11-0-5	1011 0101
B6				12-11-0-6	1011 0110
B7				12-11-0-7	1011 0111
B8				12-11-0-8	1011 1000
B9				12-11-0-9	1011 1001
BA				12-11-0-8-2	1011 1010
BB				12-11-0-8-3	1011 1011
BC				12-11-0-8-4	1011 1100
BD				12-11-0-8-5	1011 1101
BE				12-11-0-8-6	1011 1110
BF				12-11-0-8-7	1011 1111
C0		?		12-0	1100 0000
C1		A	A	12-1	1100 0001
C2		B	B	12-2	1100 0010
C3		C	C	12-3	1100 0011
C4		D	D	12-4	1100 0100
C5		E	E	12-5	1100 0101
C6		F	F	12-6	1100 0110
C7		G	G	12-7	1100 0111
C8		H	H	12-8	1100 1000
C9		I	I	12-9	1100 1001
CA				12-0-9-8-2	1100 1010
CB				12-0-9-8-3	1100 1011
CC				12-0-9-8-4	1100 1100
CD				12-0-9-8-5	1100 1101
CE				12-0-9-8-6	1100 1110
CF				12-0-9-8-7	1100 1111
D0		!		11-0	1101 0000
D1	MVN	J	J	11-1	1101 0001
D2	MVC	K	K	11-2	1101 0010
D3	MVZ	L	L	11-3	1101 0011
D4	NC	M	M	11-4	1101 0100
D5	CLC	N	N	11-5	1101 0101
D6	OC	O	O	11-6	1101 0110
D7	XC	P	P	11-7	1101 0111
D8		Q	Q	11-8	1101 1000
D9		R	R	11-9	1101 1001
DA				12-11-9-8-2	1101 1010
DB				12-11-9-8-3	1101 1011
DC	TR			12-11-9-8-4	1101 1100
DD	TRT			12-11-9-8-5	1101 1101
DE	ED (4)			12-11-9-8-6	1101 1110
DF	EDMK (4)			12-11-9-8-7	1101 1111
E0		‡		0-8-2	1110 0000
E1				11-0-9-1	1110 0001
E2		S	S	0-2	1110 0010
E3		T	T	0-3	1110 0011
E4		U	U	0-4	1110 0100
E5		V	V	0-5	1110 0101
E6		W	W	0-6	1110 0110
E7		X	X	0-7	1110 0111
E8		Y	Y	0-8	1110 1000
E9		Z	Z	0-9	1110 1001
EA				11-0-9-8-2	1110 1010
EB				11-0-9-8-3	1110 1011
EC				11-0-9-8-4	1110 1100
ED				11-0-9-8-5	1110 1101
EE				11-0-9-8-6	1110 1110
EF				11-0-9-8-7	1110 1111
F0		0	0	0	1111 0000
F1	MVO	1	1	1	1111 0001
F2	PACK	2	2	2	1111 0010
F3	UNPK	3	3	3	1111 0011
F4		4	4	4	1111 0100
F5		5	5	5	1111 0101
F6		6	6	6	1111 0110
F7		7	7	7	1111 0111
F8	ZAP (4)	8	8	8	1111 1000
F9	CP (4)	9	9	9	1111 1001
FA	AP (4)			12-11-0-9-8-2	1111 1010
FB	SP (4)			12-11-0-9-8-3	1111 1011
FC	MP (4)			12-11-0-9-8-4	1111 1100
FD	DP (4)			12-11-0-9-8-5	1111 1101
FE				12-11-0-9-8-6	1111 1110
FF				12-11-0-9-8-7	1111 1111

Kuva 39. IBM Systeemin/360 kahdeksan bitin koodi

Optisesti luettavat merkit

Eräs uusimmista tiedon tallennus- ja syöttömenetelmistä on optinen luku. Kuvassa 41 on eräitä IBM 1428 optisen lukijan hyväksymiä merkkejä. Tämän lukijan merkkivalikoima sisältää aakkoset (26 kirjainta), numerot sekä erikoismerkkejä.



Kuva 40. 7-uran ja 9-uran aakkoskoodit

IBM 1231 optinen merkkilukija tunnistaa myös tavallisella lyijykynällä tehtyjä merkintöjä, viivoja, sekä 1403 rivikirjoittimen tulostusta paperilomakkeilta. Lomakkeen koko on 8 1/2" x 11".

IBM 1285 sisältyy samaan luokkaan kuin edellisetkin laitteet, vaikka se ei varsinainen lomakkeen lukija olekaan. Sen sijaan IBM 1285 pystyy lukemaan kassa- tai laskukoneiden tulostamaa nauhaa. Sovellutusalueita löytyy mm. pankkien ja vähittäiskaupan piiristä.

IBM 1287 optinen lukija pystyy lukemaan käsin tai koneellisesti kirjoitettuja numeroita sekä eräitä kirjaimia lomakkeilta tai laskukoneen nauhalta.

No. YOUR NATIONAL BANK 1-987/210
New York, N.Y. Jan. 11, 19
PAY TO THE ORDER OF J. R. Drawee $ 56.70
Fifty-Six and 70/100 DOLLARS
SAMPLE VOID
A. B. DEPOSITOR
MARY F. DEPOSITOR
Mary F. Depositor
⑆0210⑉0987⑆ 2 20084 26 70⑈ 104 ⑇0000005670⑇
CHECK ROUTING SYMBOL | ABA TRANSIT NUMBER | ACCOUNT NUMBER | PROCESS CONTROL | AMOUNT

Magnetic Ink Characters

Enter partial payment below
MUNICIPAL WATER WORKS

Account Number	Gross Amount	Net Amount	Last Day To Pay Net
RL45332	56 01	45 98	4 30 6-

DISCOUNT TERMS: 10 DAYS

Present Reading	Previous Reading	Consumption Gals.
3255886	2369014	887

E D JONES
745 CHESTNUT ST
ANYTOWN USA

PLEASE RETURN THIS WITH YOUR PAYMENT

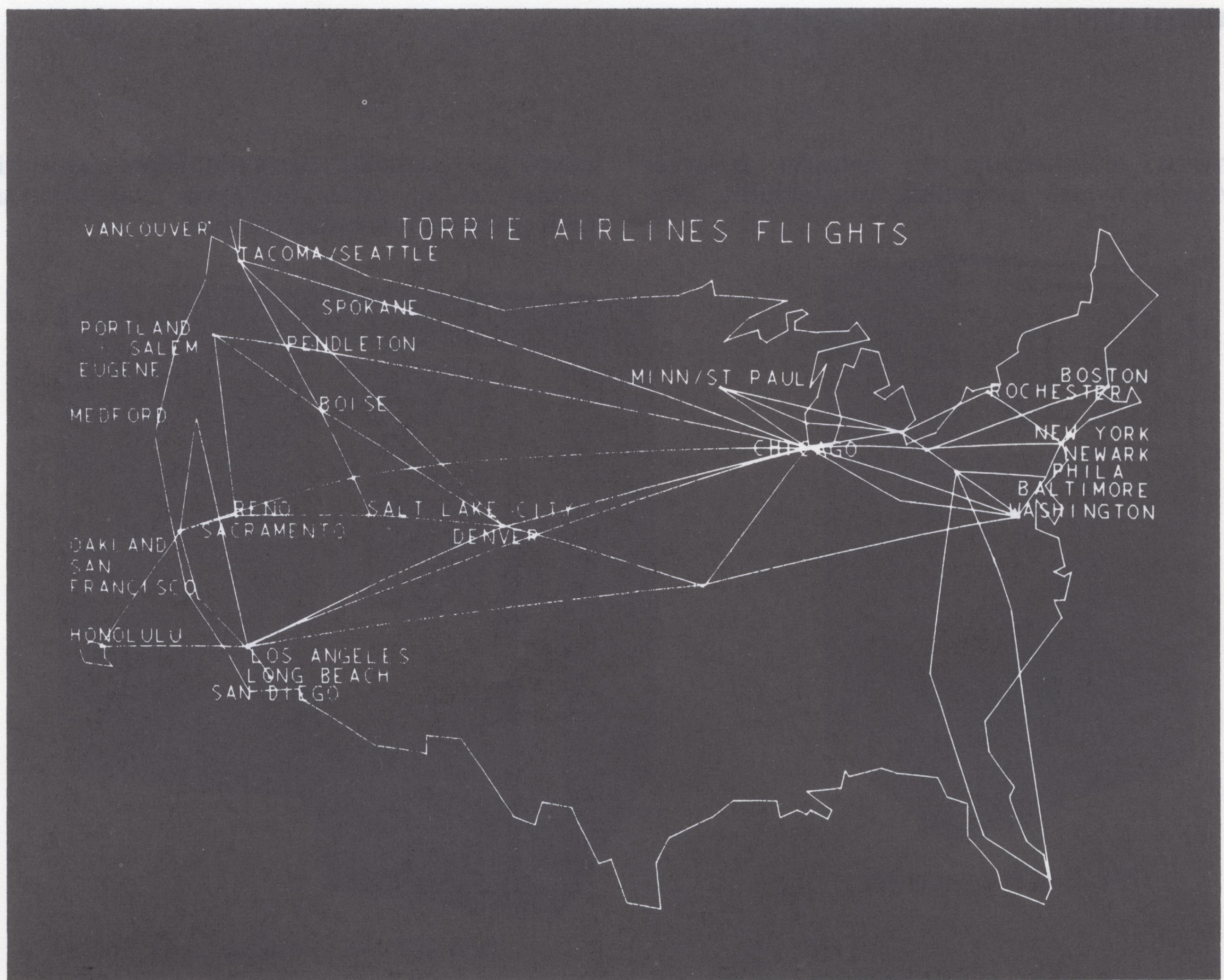
Optically Readable Characters

Kuva 41. Magneettisesti ja optisesti luettavia merkkejä

Visuaalinen tulostus

Näyttölaitteiden avulla tietokonejärjestelmän käyttäjä voi saada nähtäväkseen ja käytettäväkseen tarvitsemiaan tietoja raporttien, graafisten käyrien tms. muodossa monin verroin nopeammin kuin tavanomaisin menetelmin, esim. rivikirjoittimella tulostetun vastaavan materiaalin. Tietojen näyttö tapahtuu kuvaputkella (kuten televisiovastaanottimessa). Näyttölaitteita on saatavissa monia eri malleja, jotka eroavat toisistaan koon, nopeuden, kapasiteetin ym. ominaisuuksien suhteen. Eräs tyypillinen sovellutus on näyttölaitteen käyttö tietokonejärjestelmän valvonta- ja ohjauslaitteena (konsolina). Toinen sovellutus on esim. pankin asiakaspalveluun liittyvä kyselysysteemi, jossa asiakkaan tilin kulloinenkin saldo saadaan sitä tiedusteltaessa näkyviin tilin numeron perusteella. Tähän voidaan liittää myös tilin päivitys näyttölaitteeseen liitetyn näppäimistön avulla.

Näyttölaitteilla voidaan esittää, paitsi kirjaimia ja numeroita, taulukoita, diagrammoja tai vaikkapa



Kuva 42. IBM 2250 näyttölaitteella tulostettu kuva

karttoja (esim. kuva 42). IBM 2250 näyttölaitteen kuvaputkella on yli miljoona pistettä, joilla kullakin on X- ja Y-koordinaatteihin perustuva osoite. Näyttöalue koostuu 52 rivistä, joista jokaiselle mahtuu 74 merkkiä. Alueen koko on 12" x 12". Kuvassa 42 on näyte 2250:n tulostuksesta.

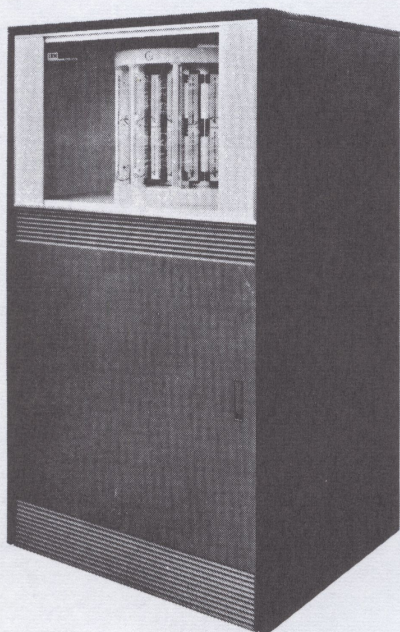
2840 ohjain, johon 2250 on kytketty, toimii tietokoneesta tulevien tietojen vastaanottajana ja puskurina 238.000 merkin sekuntinopeudella. Tietojen näyttönopeus kuvaputkella on 60.000 merkkiä tai riviä sekunnissa. Vaaka- ja pystysuoria viivoja voidaan 'piirtää' kuvaputkelle määrittämällä suoran päätepisteet. Lisälaitteen avulla voidaan viivoja piirtää mihin kulmaan tahansa. Näyttönopeus pistettä kohti on 16,8 mikrosekuntia.

IBM 2260 näyttölaitteen näyttöalue on suuruudeltaan 4 x 9" ja siihen mahtuu korkeintaan 12 riviä. Rivin pituus on 80 merkkiä.

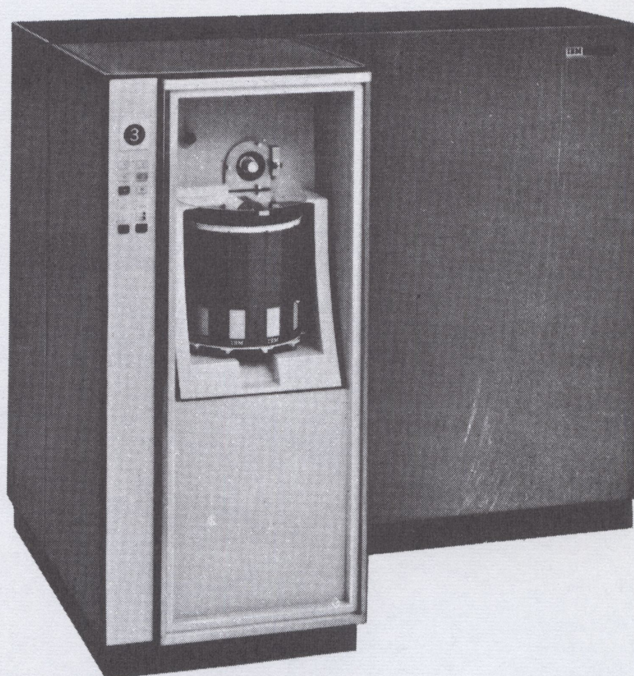
2848 ohjain, johon 2260 on kytketty, toimii tietojen vastaanottajana ja puskurina (kuten 2840) 2560 merkin sekuntinopeudella.

MUISTILAITTEET

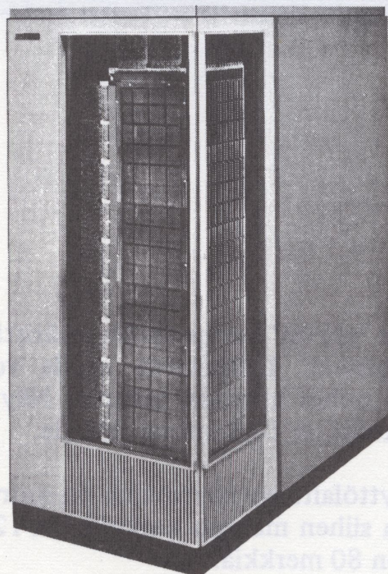
Erilaisia muistilaitteita on nykyään saatavissa hyvinkin monia: suurmuisti, rumpumuisti, levymuisti ja kennomuisti (kuva 43). Myös magneettinauhaa voidaan pitää yhtä hyvin muistilaitteena kuin syöttö- ja tulostuslaitteenakin.



IBM 2303 Rumpumuisti



IBM 2321 Kennomuisti, malli 1



IBM 2361 Suurmuisti



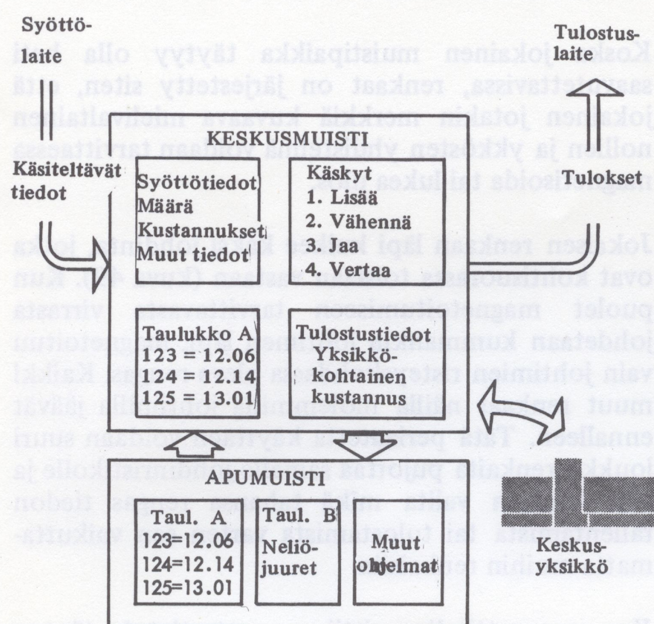
IBM 2311 Levymuisti

Kuva 43. IBM Muistilaitteita

Informaatiota voidaan tallentaa muistilaitteelle, säilyttää siellä, tai hävittää muistista miten kulloinkin tarvitaan. Informaatio voi olla itse asiassa mitä tahansa, esimerkiksi:

1. Käskyjä
2. Tiedostoja, sekä pysyvästi että tilapäisesti
3. Käyttöohjelmiin liittyviä tietoja, (taulukoita ym. jotka eivät samanaikaisesti ohjelman kanssa mahdu keskusmuistiin).

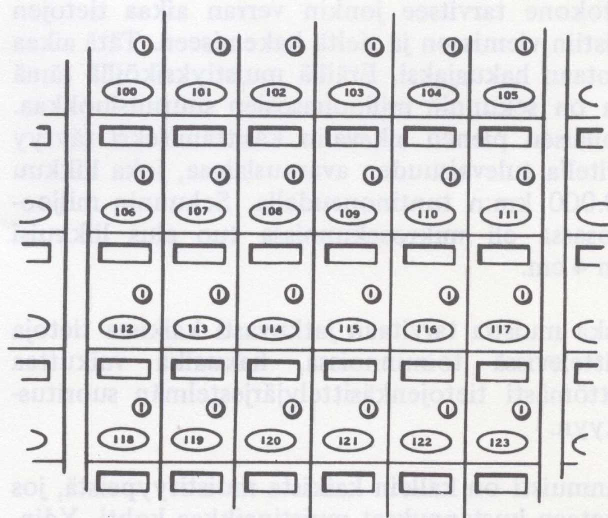
Yleisesti ottaen muistit voidaan jakaa kahteen kategoriaan, keskus- ja apumuistit (kuva 44). Keskusmuisti sisältää keskusyksikköön kuuluvan ns. ydinmuistin sekä mahdollisen suurmuistin.



Kuva 44. Kaaviokuva keskus- ja apumuistista

Käsite apumuisti sulkee sisäänsä kaikki muut muistityypit ja se voidaan edelleen jakaa kahteen tyyppiin:

1. Poimintatyyppiset. Rumpu- levy- ja kenno-muistilaitteet, joissa rekordien saanti (access) voi tapahtua suoraan, ts. tarvitsematta lukea koko tiedostoa alusta halutun rekordin löytämiseksi.
2. Peräkkäistyyppiset. Magneettinauhat, joissa halutun rekordin löytäminen edellyttää yleensä nauhakelan alusta tapahtuvaa 'etsintää'.



Kuva 45. Postitoimiston lokeroita

Pää- eli keskusmuisti ottaa tiedot lukuyksiköstä, antaa tiedot ja käskyt keskusyksikölle, saa tiedot takaisin ja voi luovuttaa ne tulostusyksikölle. Kaikkien käsiteltävien tietojen täytyy kulkea keskusmuistin kautta. Siksi sen täytyy olla niin suuri, että se pystyy vastaanottamaan kaikki tarvittavat tiedot ja niiden käsittelyssä välttämättömät käskyt.

Jokin määrätty sovellutus saattaa tehdä lisämuistin liittämisen välttämättömäksi. Tässä tapauksessa lisätään keskusmuistin kapasiteettia apu- eli sivumuistilla. Apumuisti ei ole suorassa yhteydessä keskusyksikköön ja syöttö- ja tulostusyksiköihin. Kaikkien apumuistiin menevien ja sieltä tulevien tietojen täytyy kulkea keskusmuistin kautta.

Muistia voidaan monessa suhteessa verrata postitoimiston lokeroihin (kuva 45). Jokainen lokero on merkitty omalla numerollaan. Samoin on muisti jaettu muistipaikkoihin, joista jokaisella on määrätty osoitteensa. Jokainen muistipaikka sisältää yhden tietoyksikön. Järjestelmästä riippuen tietoyksikkö voi olla merkki, numero, kokonainen tietue tai sana. Jos tieto halutaan viedä muistipaikkaan tai ottaa sieltä ulos, osoitteen täytyy olla käyttöohjelman tai ohjausohjelman tiedossa (näihin kysymyksiin palataan jäljempänä).

Jos muistipaikkaan viedään jokin tieto, niin samalla tuhoutuu sen entinen sisältö. Ulosluettaessa tieto sen sijaan jää muuttumattomana muistipaikkaan. Siksi voidaankin muistissa olevia tietoja käyttää useampaan kertaan eli tieto voidaan monistaa.

Tietokone tarvitsee jonkin verran aikaa tietojen muistiin viemiseen ja sieltä hakemiseen. Tätä aikaa sanotaan hakuajaksi. Eräillä muistiyksiköillä tämä aika on sekunnin miljoonasosien suuruusluokkaa. Tuollaisen pienen aikavälin käsittämiseksi täytyy kuvitella tulevaisuuden avaruuslaivaa, joka liikkuu 160.000 km:n tuntinopeudella. Sekunnin miljoonasosassa eli mikrosekunnissa tuo alus liikkuisi noin 4 cm.

Koska muistia tarvitaan jatkuvasti kaikissa tietoja käsittelevissä toiminnoissa, hakuaika vaikuttaa välittömästi tietojenkäsittelyjärjestelmän suorituskykyyn.

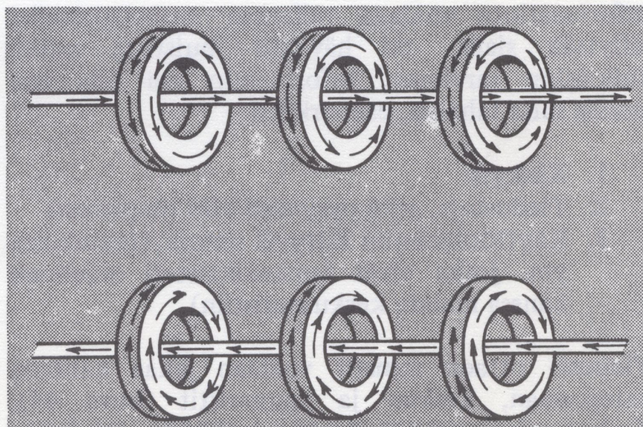
Ydinmuisti on kallein kaikista muistityypeistä, jos lasketaan kustannukset muistipaikkaa kohti. Ydinmuistilla on kuitenkin lyhin hakuaika, joten se silti saattaa olla taloudellisin muistityyppi. Rumpumuistien etuna on välittömien kustannusten pienuus, mutta rumpumuistin nopeus on myös pienempi. Useimmat levymuistit ovat vielä hitaampia kuin rumpumuistit, mutta sen sijaan niiden muistikapasiteetti on miljoonia positioita. Kapasiteetiltaan suurin muistilaite on kennomuisti, johon mahtuu 400 miljoonaa (tai 800 miljoonaa pakattua) numeroa.

YDINMUISTI

Magneettiytimeksi sanotaan pienen pientä ferromagneettisesta aineesta tehtyä rengasta, jonka läpimitta voi olla muutama tuuman sadasosa. Jokainen rengas puristetaan rautaoksidijauheen ja joidenkin lisäaineiden seoksesta ja poltetaan.

Pienen tilantarpeen lisäksi (seikka, joka on tietokoneita rakennettaessa merkittävä tekijä) magneettiytimellä on se tärkeä ominaisuus, että se voidaan magnetoida muutamassa sekunnin miljoonasosassa. Magnetointi säilyy käytännöllisesti katsoen ikuisesti, ellei sitä tarkoituksellisesti muuteta.

Jos magneettiytimet pujotetaan johtimelle niinkuin helmet lankaan ja kyllin voimakas sähkövirta johdetaan johtimen läpi, renkaat magnetoituvat (kuva 46). Virran suunta määrää renkaan polariteetin eli magneettisen tilan (kuva 47). Virran suunnan kääntäminen muuttaa magneettisen tilan (kuva 48). Näin ollen voidaan näitä kahta tilaa käyttää esittämään bittejä 0 ja 1, plus tai miinus, kyllä tai ei. Koneelle tämä on binäärisen muistijärjestelmän perusta.



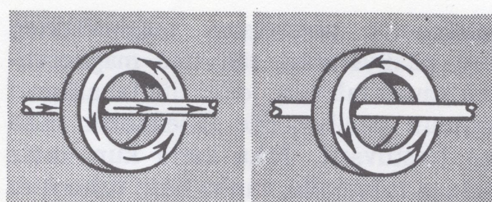
Kuva 46. Magneettiytimen polariteetti

Koska jokainen muistipaikka täytyy olla heti saavutettavissa, renkaat on järjestetty siten, että jokainen jotakin merkkiä kuvaava mielivaltainen nollien ja ykkösten yhdistelmä voidaan tarvittaessa magnetisoida tai lukea ulos.

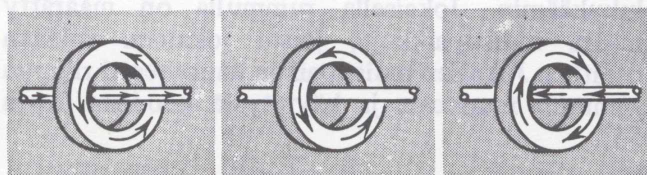
Jokaisen renkaan läpi kulkee kaksi johdinta, jotka ovat kohtisuorassa toisiaan vastaan (kuva 49). Kun puolet magnetoitumiseen tarvittavasta virrasta johdetaan kummankin johtimen läpi, magnetoituu vain johtimien risteyskohdassa oleva rengas. Kaikki muut renkaat näillä molemmilla johtimilla jäävät ennalleen. Tätä periaatetta käyttäen voidaan suuri joukko renkaita pujottaa samalle johdinristikolle ja silti voidaan valita mikä tahansa rengas tiedon tallentamista tai tulostamista varten sen vaikuttamatta muihin renkaisiin.

Kun magneettiydinmuistiin on saatu tietoja, täytyy keksiä keino, jonka avulla ne voidaan tarvittaessa saada käytettäväksi. Aikaisemmin jo selitettiin, että renkaaseen voidaan saada aikaan määrätty magneettinen polariteetti, jos virran annetaan kulkea johtimen läpi. Virta ei itse asiassa ole vakio, vaan se lähetetään pulsseina. Sellainen pulssi muuttaa aina virran suunnasta riippuen renkaan joko positiiviseen tai negatiiviseen tilaan.

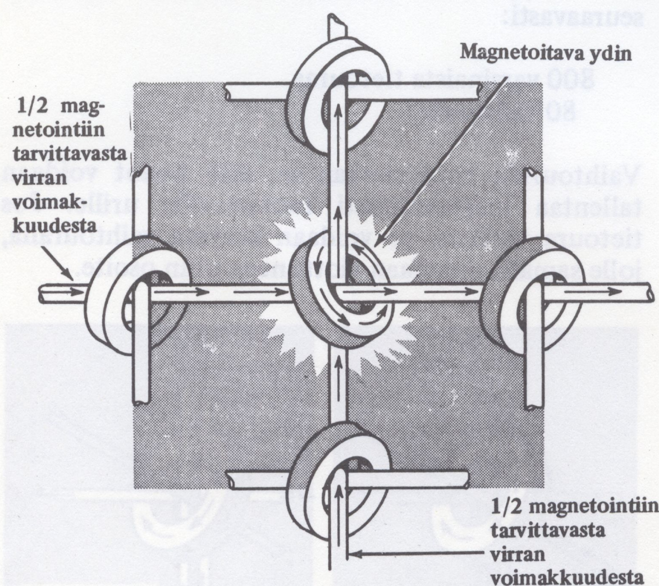
Jos pulssi muuttaa renkaan magneettisen tilan, tämä äkillinen muutos indusoi virran kolmanteen, samoin renkaan läpi kulkevaan johtimeen (kuva 50). Tässä tuntojohtimessa kulkevan signaalin avulla kone voi tuntea, sisälsikö ydin 1:n vai 0:n. Kokonaista ferriittirengastasoa varten tarvitaan vain yksi tuntojohdin, koska jokaisessa tasossa testataan aina vain yhden renkaan magneettista tilaa kerrallaan. Johdin kulkee sen tähden kaikkien tason renkaitten läpi (kuva 51).



Kuva 47. Ytimen magnetointi

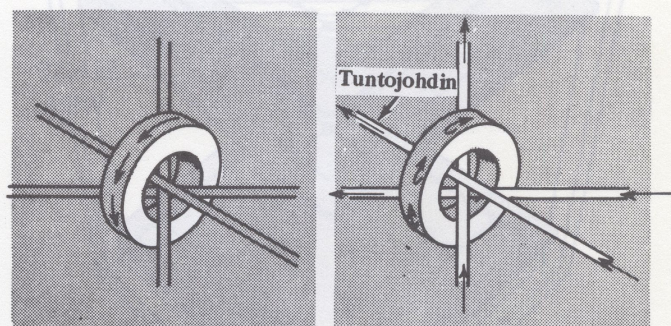


Kuva 48. Ytimen magneettisen tilan muuttaminen



Kuva 49. Magnetoitavan ytimen valinta

On kuitenkin huomattava, että kaikki renkaat asetetaan nollaan, kun niiden sisältämät tiedot luetaan muistista. Ulosluku vaikuttaa sikäli häiritsevästi, että ykköstä luettaessa rengas muuttuu nollaksi. Tietojen säilyttämiseksi ennallaan täytyy koneen sen tähden palauttaa ykkönen niihin renkaisiin, jotka sen alun perin sisälsivät. Niiden renkaiden, jotka aikaisemminkin sisälsivät nollan, täytyy edelleen jäädä nolliksi.

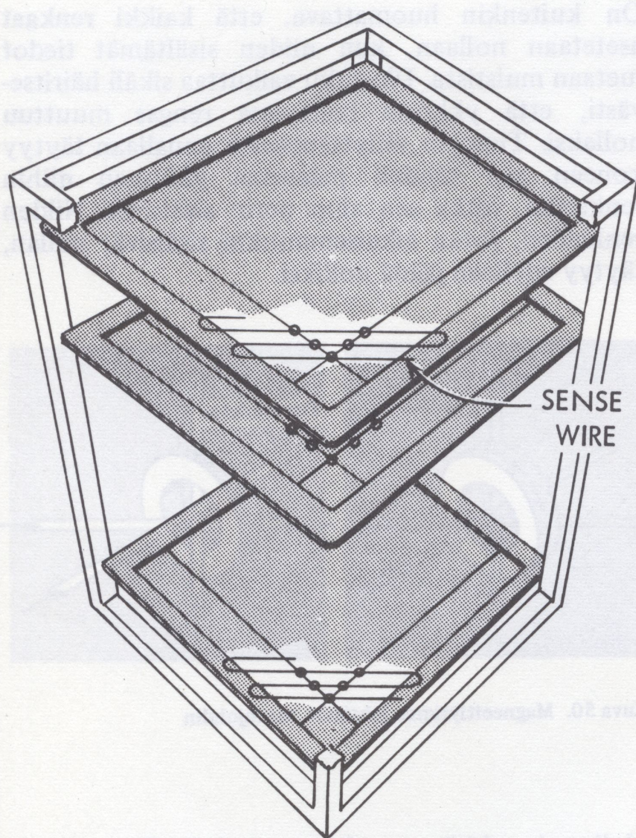


Kuva 50. Magneettiytimen johtimet, tuntojohdin

Nollan ja ykkösen palauttamiseksi alkuperäiseen muotoon yrittää tietokone kirjoittaa ykkösen kaikkin niihin muistipaikkoihin, joista on sitä ennen luettu tiedot. Samanaikaisesti estopulssi mitätöi kirjoituksen niissä renkaissa, jotka alun perin sisälsivät nollan. Tämä estopulssi kulkee neljännessä johtimessa ja vaimentaa kirjoituspulssin toisessa magnetoimiseen käytettävässä johtimessa. Samoin kuin tuntojohdin kulkee myös estojohdin tason jokaisen renkaan läpi (kuva 52).

Magneettiydinmuistin täydellinen selvittäminen menee tämän kirjan puitteiden ulkopuolelle, mutta muutamat perustiedot ferriittirengasmuistin toiminnasta ovat kuitenkin hyödyllisiä kaikkien tällaista muistia käyttävien tietojenkäsittelyjärjestelmien toimintojen ymmärtämiseksi. Joissakin ydinmuisteissa tunto- ja estotoiminnot hoidetaan yhdellä johtimella, joka virran kulkiessa yhteen suuntaan toimii tuntojohtimena ja virran kulkiessa vastakkaiseen suuntaan toimii estojohtimena.

Kehittyneemmissä järjestelmissä, kuten esim. IBM 2361 (kuva 10 ja 43), käytetään vain kahta johdinta ja toimintakin perustuu aivan eri periaatteille niissä käytetyt ratkaisut ovat tehokkaita ja nopeita, vaikka varsinainen bittien käsittely onkin samanlaista kuin edellä on kuvattu.



Kuva 51. Ydintason tuntojohdin

RUMPUMUISTI

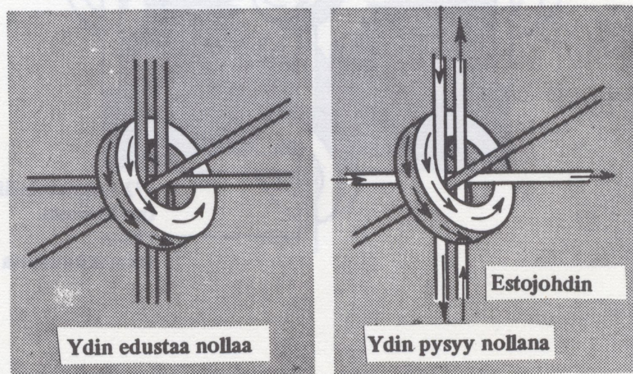
Rumpumuistitoiminnoissa käytetään kahta periaatetta. Ensimmäinen ja alkuperäinen periaate oli käyttää rumpumuistia suurikapasiteettisena, välitöntä hakuperiaatetta käyttävänä muistilaitteena. Tässä sovellutuksessa sitä periaatteellisesti käytettiin sellaisten tietojen tallentamiseen, joita tarvittiin toistuvasti toiminnan aikana (aktuaaritaulukot, logaritmitaulukot jne) tai päämuistin alaisena apumuistina. Toinen ja myöhäisempi rumpumuistien sovellutus on käyttää sitä hakumuistien alijärjestelmissä tallentamaan ohjelmia, ohjelmien modifioimistietoa ja väliaikaisena muistina suuren käyttötiheyden omaavissa, rajoitetun tietomäärän sisältävissä hakutoiminnoissa.

Magneettirumpu on vakionopeuksinen pyörivä sylinteri, jonka uloin kerros on päällystetty magneettisella aineella. Kun alue tätä magneettista ainetta joutuu magneettikenttään, se magnetoituu. Kun magneettikenttä poistetaan, magneettinen 'jälki' rummun päällysteessä säilyttää magneettisuutensa miten kauan tahansa, jollei sitä tahallisesti hävitetä. Rummun päällysteelle tallennettu tieto voidaan lukea toistuvasti. Joka kerta kun uutta tietoa tallennetaan rummulle, vanha tieto tuhoutuu automaattisesti. Tieto tallennetaan pyörivälle rummulle ja luetaan siitä luku-kirjoituspäillä, jotka ovat hyvin lähellä rummun pintaa. Luku-kirjoituspäät sisältävät hienosta langasta käämittyjä käämejä, jotka on käämitty pienten magneettiydinten ympärille (kuva 53). Magneettirummun päällys magnetoituu, kun virtaimpulssi lähetetään luku-kirjoituspään kirjoituskäämiin. Rummulle tallennettu tieto voidaan käyttää silloin, kun se alittaa lukukäämin. Jokaisella rummulla on määrätty määrä muistipaikkoja, joista jokainen voidaan osoittaa. Jokaisen muistipaikan kapasiteetti riippuu rummun muodosta ja käytetystä koodista (kuva 55).

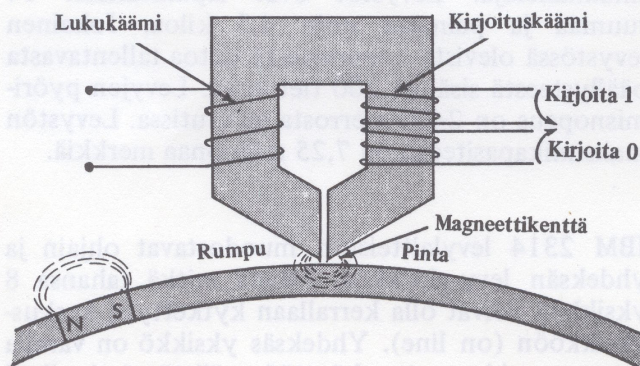
Esimerkiksi rumpumuistista ja sen toiminnasta voidaan ottaa IBM 2303 rumpumuisti, jossa muistin muodostavat pystysuunnassa oleva rumpu ja tarvittavat virtapiirit. Rumpu, joka on päällystetty magneettisella kalvolla, pyörii noin 3500 kierroksen minuuttinopeudella. Rummun pinta on jaettu uriin, joilla kullakin on oma osoitteensa. Tietojen tallennus tapahtuu näille urille, joita on 880 kappaletta, jakautuen kahteen ryhmään seuraavasti:

800 varsinaista tietouraa
80 vaihtouraa

Vaihtourilla varmistetaan se, että tiedot voidaan tallentaa magneettisesti luotettaville urille. Jos tietoura viottuu, se voidaan korvata vaihtouralla, jolle samalla annetaan viottuneen uran osoite.



Kuva 52. Ytimen estojohdin



Kuva 53. Tiedon tallennus magneettirummulla

Rumpumuistissa on luku-kirjoituspää jokaista uraa varten. Luku- kirjoituspäät on sijoitettu kiinteästi 20 pystysuoraan rumpua ympäröivään telineeseen. Kussakin telineessä on 40 luku-kirjoituspäätä. Tarvittaessa huoltoteknikko voi helposti muuttaa viallisen uran luku-kirjoituspään vaihtouralle.

Luku-kirjoituspää sisältää pienen johtimella päällystetyn magneettisydämen. Kirjoitusoperaatiossa magneettisydän muuntaa tietokoneesta tulevat sähköiset signaalit magneettisiksi voimaviivoiksi, magneettikentäksi, jolloin määrätyt rummun pinnalla olevat pisteet magnetoituvat. Lukuoperaatiossa toiminta on päinvastainen: rummun pinnalla olevat magnetoidut pisteet muodostavat magneettisen kentän, jonka luku-kirjoituspää muuntaa sähköisiksi signaaleiksi ja siirtää tietokoneen keskusyksikköön.

Tiedon saantiajat
(Access times)

Rumpumuisteissa tiedon saanti muodostuu kahdesta komponentista (vrt. levymuistit, joissa komponentteja on kolme). Komponentit ovat mekaaninen ja elektroninen.

LEVYMUISTI

Levymuistit, kuten rumpumuistitkin, antavat mahdollisuuden tietojen tallentamiseen ja tietojen saantiin sekä perättäis- että hajajärjestyksessä.

On mahdollista kohdistaa haku johonkin määrättyyn tietoalueeseen tarvitsematta tarkistaa järjestelmällisesti kaikkea tallennettua tietoa. Magneettinauhoja käytettäessä ei ole tätä mahdollisuutta: haku täytyy aloittaa nauhakelan alusta ja sitä täytyy jatkaa tietueiden läpi, kunnes haluttu tietoalue löytyy.

Esimerkkinä poimintamuistisovellutuksesta järjestelmällistä hakua käyttävään verrattuna olettakaamme, että haluamme hakea yhden sanan täydellisestä sanastosta. Jos sanaston sisältö olisi tallennettu magneettinauhalle, koko sanaston koneellinen lukeminen veisi aikaa noin kaksi minuuttia. Magneettinauhan järjestelmällistä hakumenetelmää käytettäessä kuluisi sanan hakemiseen ja lukemiseen keskimäärin yksi minuutti. Ihmisen käyttäessä sanastoa hakuajaksi tulisi noin 15 sekuntia/sana, koska ihminen alkaisi etsimisen sanaston oikeasta osasta, toisin sanoen määrätystä kirjaimesta eikä koko sanaston alusta. Tällä periaatteella, jossa haku kohdistuu kokonaisuuden osaan, poimintamuisti suorittaisi sanan haun muutamassa sekunnin tuhannesosassa.

Poimintamuistia käyttävän tietojenkäsittelyjärjestelmän tuoma suuri hakunopeus sallii käyttäjän pitää ajan tasalla olevia tiedostoja ja tehdä usein toistuvia viittauksia tallennettuun tietoon.

Magneettilevy on ohut metallinen levy, joka on päällystetty molemmilta puolilta magneettisella aineella. Levyt on asennettu pystysuoralle akselille. Ne on erotettu toisistaan pienellä välillä luku-kirjoituspään liikkumisen vuoksi. Akseli pyörii ja pyörittää samalla levyjä (kuva 54).

Tieto tallennetaan magneettisina pisteinä samankeskisiin uriin levyn kummallekin puolelle. Muutamissa yksiköissä on 500 uraa levyn kummallakin puolella käytettävissä tietojen tallentamiseen. Urat haetaan lukua ja kirjoitusta varten sijoittamalla luku-kirjoituspäät pyörivien levyjen väliin.

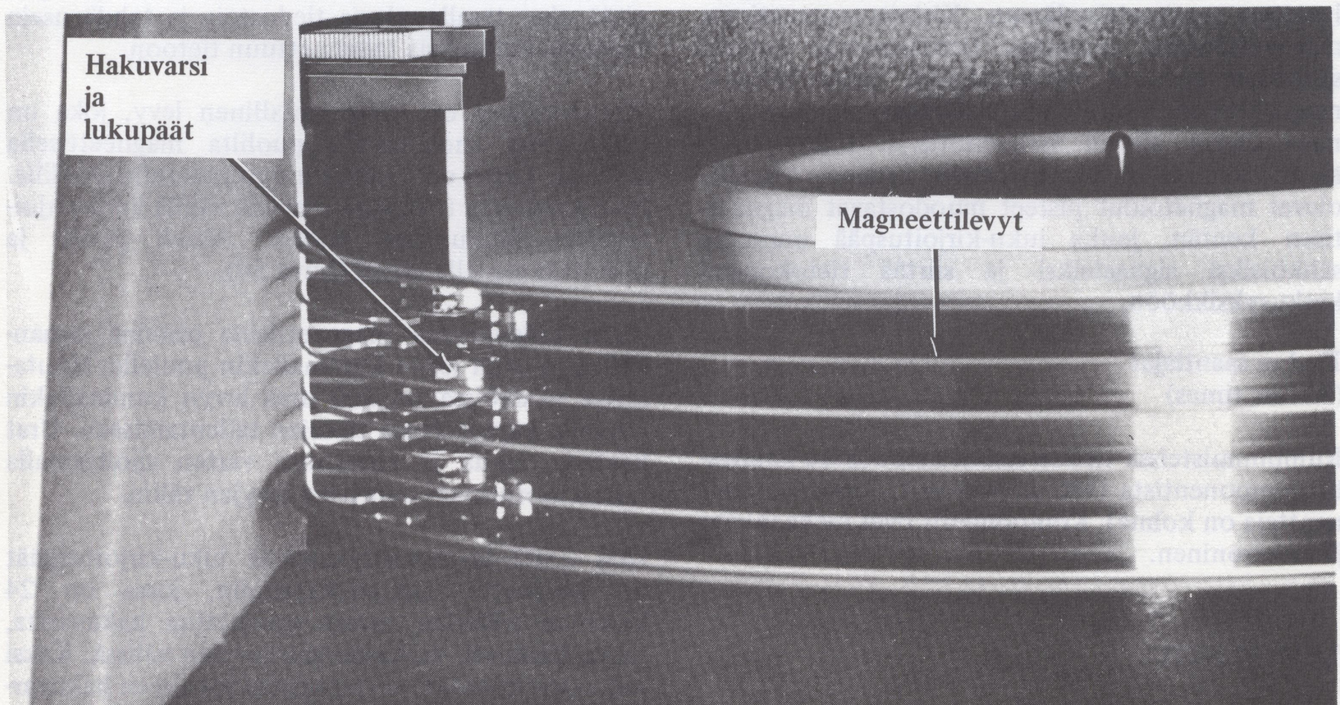
IBM 2302 levymuistiyksikössä luku-kirjoituspäät on asennettu hakumekanismiin, jossa on 24 kamman piikkien tavoin asennettua hakuvartta, jotka liikkuvat vaakasuoraan levyjen välissä. Kaksi luku-kirjoituspäätä on asennettu jokaiseen hakuvarteen. Toinen pää huolehtii ylemmän levyn alapinnasta samalla aikaa kun toinen pää huolehtii alemman levyn yläpinnasta. On siis mahdollista lukea tai kirjoittaa levyn molemmille sivuille.

Magneettilevyn päällyste voidaan käyttää toistuvasti. Joka kerta kun uutta tietoa tallennetaan urille, vanha tieto tuhoutuu niiltä osin, mihin uutta tietoa tallennetaan. Tallennettu tieto voidaan käyttää niin usein kuin halutaan, tieto pysyy levyn riville tallennettuna niin kauan kunnes sen päälle kirjoitetaan.

IBM 2302 levymuistiyksikkö sisältää yhden tai kaksi levymodulia. Kumpikin moduli koostuu 25:stä magneettisesti päällystetystä levystä, läpimitta kaksi jalkaa, sekä hakumekanismista. Levyn kummallakin puolella on 492 uraa. Levyt on asennettu puolen tuuman päähän tosistaan pyörivälle pystysuoralle akselille.

IBM 2311 levymuistiyksikkö on toimintaperiaatteiltaan samanlainen kuin 2302 paitsi että 2311 levymuistiyksikkö (kuva 54) käyttää vaihdettavia levystöjä. Levystöön kuuluu kuusi levyä. Levystöt voidaan helposti poistaa yksiköstä ja tallentaa levystöjen kirjastoon samalla tavalla kuin magneettinauhakeloja. Levystöt ovat läpimitaltaan 14 tuumaa ja painavat noin 4,5 kiloa. Jokainen levystössä olevista kymmenestä tietoa tallentavasta päällysteestä sisältää 200 tietouraa. Levyjen pyörimisnopeus on 2400 kierrosta minuutissa. Levystön maksimikapasiteetti on 7,25 miljoonaa merkkiä.

IBM 2314 levylaitteiston muodostavat ohjain ja yhdeksän levy-yksikköä. Näistä mitkä tahansa 8 yksikköä voivat olla kerrallaan kytkettynä keskusyksikköön (on line). Yhdeksäs yksikkö on varalla joten se voidaan ottaa käyttöön välittömästi mikäli joku muista yksiköistä on korjauksen tai huollon tarpeessa. Yksiköissä käytettävät levystöt ovat samankaltaisia vaihdettavia levystöjä kuin mitä IBM 2311 levy-yksiköissäkin käytetään. 2314:n levystöt ovat kuitenkin suurempia sisältäen 11 levyä, jolloin tietojen tallennukseen on käytettävissä 20 levypintaa. Jokaisella levypinnalla on 200 tietouraa. Yhden levystön maksimikapasiteetti on 29,18 miljoonaa tavua.



Kuva 54. IBM 2311 levymuistiyksikkö, levystö ja hakuvarsi

Kennomuisti

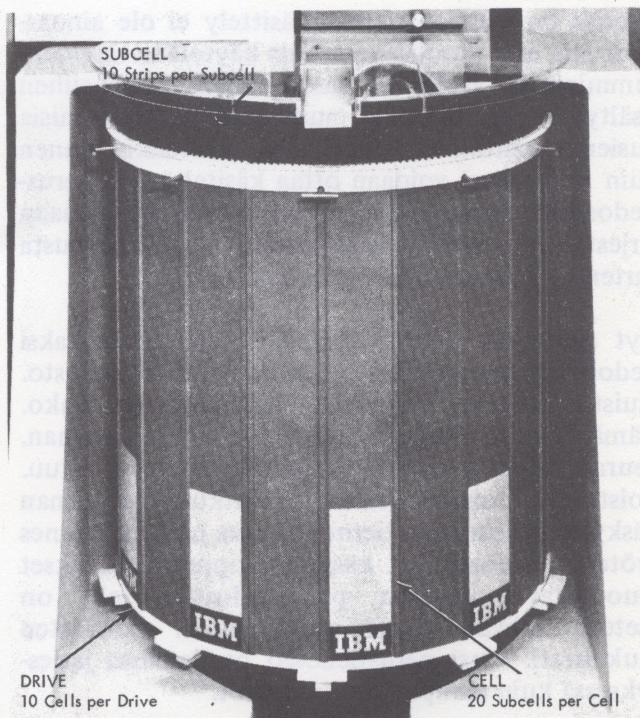
Tietokenno (ks. kuva 55) sisältää kaksisataa ohutta magneettista, mitoiltaan noin 2" x 12" suuruista liuskaa. Kymmenen liuskaa muodostaa kennon lohkon (sub-cell). Kaksikymmentä lohkoa vuorostaan muodostaa tietokennon. Kullakin kennossa olevalla liuskalla on oma osoitteensa ja tietojen haku ja tallennus kohdistuvatkin juuri liuskoihin, tai tarkasti sanoen kullakin liuskalla olevaan 200 tietouraan.

Kennomuistiyksikkö voi sisältää 10 kennoa. Luvun ja kirjoituksen yhteydessä haluttu kenno siirtyy hakumekanismin kohdalle. Kennot ovat vaihdettavia. Kuhunkin kennoon (kuva 56) mahtuu 40 miljoonaa tavua.

Kennomuisti tarjoaa taloudellisen mahdollisuuden poimintatyyppisen muistikapasiteetin lisäämiseksi - sen avulla voidaan saada linjaan (on line) huomattavasti enemmän tietoja kuin muilla vastaavilla laitteilla. Kennomuistin kapasiteettihan on 400 miljoonaa tavua.



Kuva 55. IBM 2321 Kennomuisti



Kuva 56. IBM 2321 kennomuisti, kenno ja kennolohko (sub-cell).

MUISTI JA TIETOJENKÄSITTELYMENETELMÄT

IBM-tietojenkäsittelyjärjestelmät käyttävät kahta tietojenkäsittelymenetelmää, eräkäsittelyä ja suorasaantikäsittelyä (ks. kuvaa 57). Käytettävä menetelmä riippuu sovellutuksesta.

Molemmissa tapauksissa kaikki yksittäiseen sovellutukseen liittyvä tieto on tiedostoissa.

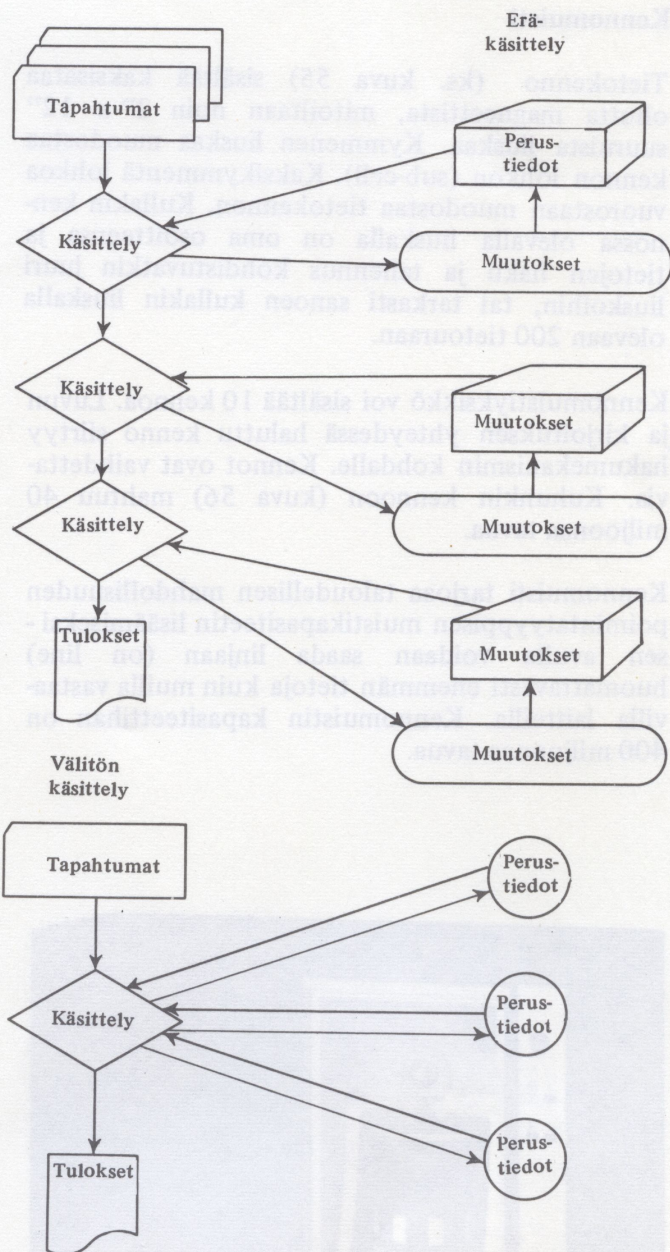
Eräkäsittelyssä tiedostot ovat tallennettuina tietokoneen ulkopuolelle, tavallisesti magneettinauhoille, ja ne on järjestetty ennalta määrättyyn järjestykseen. Tieto voi koskea tavaraluetteloita, tuloja ja menoja, palkkoja jne. Tiedostot koostuvat tietueista, joista jokainen sisältää yksityisen osan täydelliseen kuvaamiseen tarvittavat tiedot. Järjestyksen voi määrätä osanumero, nimi, tilinumero tai henkilönumero, mutta määrättyyn sovellutukseen liittyvien kaikkien tiedostojen täytyy olla samassa järjestyksessä, siis saman perusteen mukaan järjestettyinä.

Useissa tapauksissa tietojenkäsittely ei ole ainoastaan sitä, että tietojaksojen osia käytetään taseiden, summien ja ansioiden laskemiseen, vaan siihen sisältyy myös lisäyksiä, muutoksia tai poistamisia uusien tapahtumien esiintyessä. Kuitenkin ennen kuin muutokset voidaan ottaa käsiteltäviksi perustiedoston kanssa, ne täytyy järjestää samaan järjestykseen kuin perustiedosto. Tätä tarkoitusta varten ne kootaan sopiviin ryhmiin.

Nyt syötetään tietojenkäsittelyjärjestelmään kaksi tiedostoa, perustiedosto ja tapahtumatiedosto. Muistiin luetaan kerrallaan yksi tietue tai lohko. Nämä tiedot käsitellään ja tulos tulostetaan. Seuraava lohko luetaan ja sama käsittely toistuu. Toistuvien toimintojen sarja jatkuu ohjelman käskyjen ohjaamana, tietue toisensa perään, kunnes syötetyt tiedostot on käsitelty loppuun. Tulokset muodostavat korjatun perustiedoston, joka on asetettu päivän tasalle esiintyneiden muutosten mukaisesti. Uusi perustiedosto on samassa järjestyksessä kuin alkuperäiset tiedostot.

Käsittelyn sivutuotteena voidaan saada myös tulostusta, joka voi sisältää tietoja vitkastelevista maksajista, pankkisiirroista, palkoista jne. Joka tapauksessa kaiken tulostuksen järjestys on sama kuin syötetyn tiedon järjestys. Peräkkäiskäsittelyssä tieto on muistissa hetkellisesti. Sen tähden muistiyksikön tarvitsee olla vain niin suuri, että se voi sisältää ohjelmassa olevat käskyt ja suurimman käsiteltävän tietoelementin.

Kun käytetään suorasaantikäsittelyä, tiedoston sisältöön vaikuttavat muutokset syötetään tietokoneeseen satunnaisesti, silloin kun niitä esiintyy. Tässä tapauksessa tietokone hakee vastaavan jakson tai tiedon tiedostosta ja korjaa sitä muutoksen ilmoittamalla määrällä. Summat ja saldot pidetään aina kunnossa ja ne voidaan tarvittaessa käyttää tulostukseen. Muutoksia ei tarvitse ryhmittää eikä niitä tarvitse lajitella ennen käsittelyä.

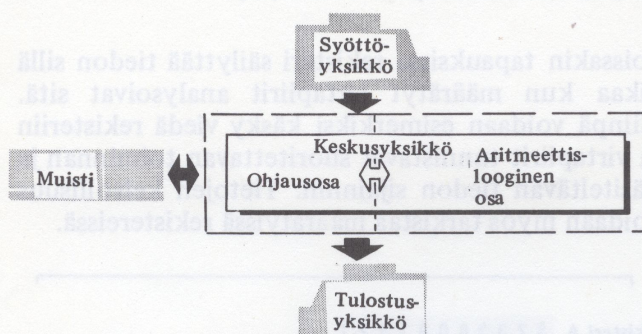


Kuva 57. Eräkäsittely (Batch Processing) ja välitön käsittely (on line processing)

KESKUSYKSIKKÖ

Keskusyksikkö ohjaa ja valvoo koko tietojenkäsittelyjärjestelmää ja suorittaa tiedoille varsinaiset aritmeettiset ja loogiset toiminnat. Keskusyksikkö käsittää kaksi toiminnaltaan erilaista osaa: ohjausosan ja aritmeettis-loogisen osan (kuva 58).

Ohjausosa johtaa ja järjestää kaikki ohjelman vaatimat toiminnat. Tähän kuuluu syöttö- ja tulostuslaitteiden ohjaus, tietojen muistiin vienti ja sieltä haku, sekä tietojen kulku muistin ja aritmeettis-loogisen osan välillä. Ohjausosa takaa koko tietokoneen automaattisen ja mielekkään toiminnan.



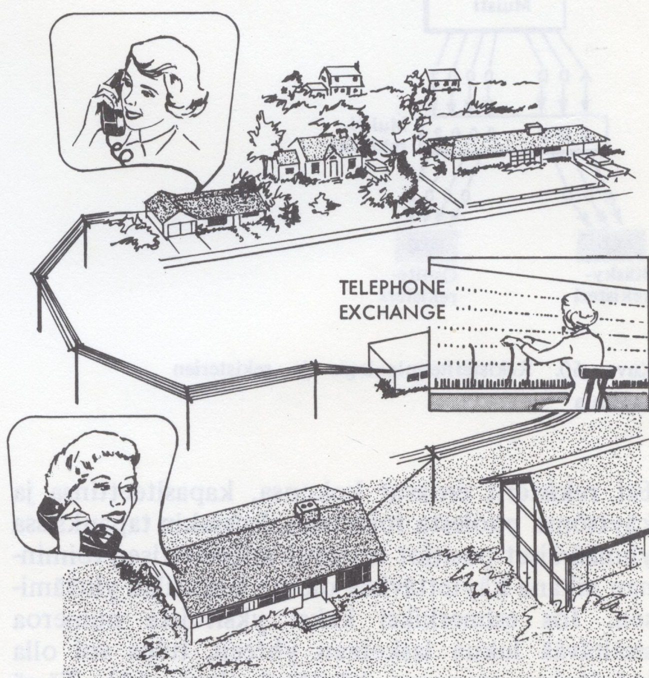
Kuva 58. Tietojenkäsittelyjärjestelmän keskusyksikkö

Ohjausosaa voidaan monessa suhteessa verrata puhelinkeskukseen. Tietojen välitystä varten on jo olemassa kaikki mahdolliset tiet, samoin kuin kaikkien keskukseen liitettyjen puhelimien välillä on yhteysjohdot (kuva 59).

Puhelinkeskus valvoo laitteita, jotka siirtävät äänen puhelinkoneesta toiseen, hälyttävät, sulkevat virtapiirejä ja taas avaavat ne. Yhteys kahden puhelimen välillä saadaan aikaan tarkoituksenmukaisella ohjauksella itse laitteessa. Tietokoneessa joudutaan yhden käskyn toteuttamiseksi avaamaan ja sulkemaan monia teitä ja portteja. Ohjausosa voi käynnistää tai pysäyttää syöttö- ja tulostuslaitteen, kytkeä signaalilaitteen toimimaan tai poistaa sen toiminnasta, kelata magneettinauhan takaisin ja ohjata laskutoimituksia.

Joissakin Systeemin/360 malleissa osan tästä keskuksesta muodostaa ohjausjärjestelmä nimeltä lukumuisti (read-only storage) eli mikro-ohjelma, jonka muodostavat koneen käskykoodien määrittämien toimintojen toteuttamiseen tarvittavat painetut virtapiirit. Tässä 'muistissa' sijaitsevat myös emulaattorikytkennät, joiden avulla Systeemillä/360 voidaan toteuttaa myös muita tietokoneita varten kirjoitettuja ohjelmia.

Aritmeettis-loogisessa osassa on virtapiirit, jotka suorittavat aritmeettiset ja loogiset toiminnat. Aritmeettisia toimintoja ovat laskutoimitukset, siirrot, tulosten varustaminen algebrallisilla etumerkeillä, pyöristys, vertailu jne. Looginen osa tutkii erilaisia käsittelyn aikana ilmaantuvia ehtoja ja muuttaa käsittelyjärjestystä testaustulosten mukaan.



Kuva 59. Puhelinkeskusjärjestelmä

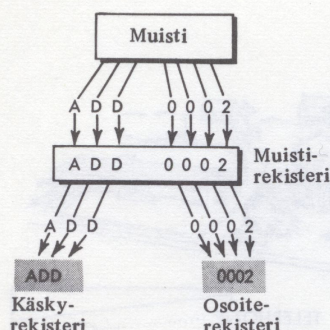
TOIMINTAYKSIKÖT

Rekisteri

Rekisteri on laite, joka ohjevirtapiirien johdolla ottaa vastaan tiedot, säilyttää ne ja siirtää jollekin toiselle paikalle. Sen elektronisia komponentteja ovat magneettiytimet tai transistorit.

Rekistereiden nimet ovat toiminnan mukaisia. Laskulaite kerää tuloksia, kertoja-osamäärärekisteri säilytttää kertojan tai osamäärän. Muistirekisteri sisältää tiedot, jotka haetaan muistista tai viedään sinne. Osoiterekisteri säilyttää muistipaikan tai laitteen osoitteen, käskyrekisteri säilyttää käskyn, jota juuri suoritetaan (kuva 60).

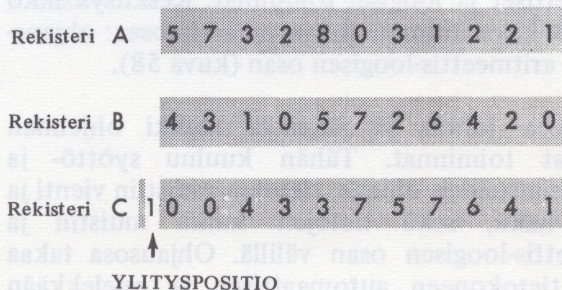
Systeemissä/360 on yleisrekistereitä, joita voidaan käyttää moniin tarkoituksiin. Rekistereiden sisältönä voi olla osoite (kanta- ja indeksirekisterit) tai mikä tahansa loogisesti tai aritmeettisesti käsiteltävä tieto.



Kuva 60. Rekisteriterminologia ja rekisterien toiminta

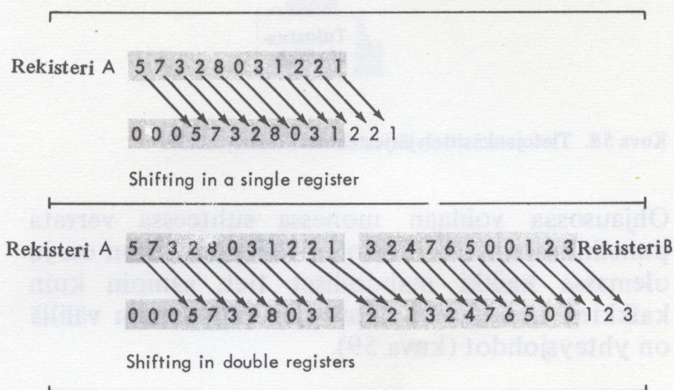
Eri rekisterit eroavat kokonsa, kapasiteettinsa ja käyttönsä puolesta toisistaan. Joissakin tapauksissa ylimääräiset positiot toteavat aritmeettisen toiminnan aikana käytettävissä olevan laskutilan ylittämisen. Jos esimerkiksi kaksi yksitoista numeroa sisältävää lukua lasketaan yhteen, tulos voi olla kaksitoista numeroa käsittävä (kuva 61). Tässä kuvassa rekisteri A sisältää toisen yhteenlaskettavan ja rekisteri B toisen.

Tekijät lasketaan yhteen ja summa sijoitetaan rekisteriin C, jossa tiedon ilmestyminen ylitysposi-tioon merkitsee ylitystilannetta. Joittenkin rekisterien sisällöt voidaan siirtää rekistereiden sisällä vasemmalle tai oikealle, usein voidaan myös suorittaa rekistereiden välillä tapahtuva siirto. Tyhjentyneet positiot täytetään nollilla ja rekisteristä ulos joutuvat numerot menetetään.



Kuva 61. Yhteenlaskusta johtuva alueen ylitys

Joissakin tapauksissa rekisteri säilyttää tiedon sillä aikaa kun määrätyt virtapiirit analysoivat sitä. Niinpä voidaan esimerkiksi käsky viedä rekisteriin ja virtapiirit tunnistavat suoritettavan toiminnan ja käsiteltävän tiedon sijainnin. Tietojen kelvollisuus voidaan myös tarkistaa määrätyissä rekistereissä.



Kuva 62. Rekisterisiirrot (shifting) yhden ja kahden rekisterin puitteissa

Tärkeimpiin rekistereihin, varsinkin tietojen kulkua ja osoitteitusta koskeviin rekistereihin on liitetty pienet hehku- ja neonlamput. Nämä lamput ovat ohjaustaulussa (kuva 63) ja ne voivat osoittaa rekisterien sisältöjä ja erilaisia ohjelmatilanteita.

Laskulaite

Laskulaite on verrattavissa rekisteriin ja se suorittaa tavallisesti samoja toimintoja. Lisäksi voidaan sen sisältöä lisätä tai vähentää jollakin määrällä. Laskulaitteen toiminta riippuu sen rakenteesta ja

lisäksi tehtävästä, johon sitä tietokoneessa käytetään. Samoin kuin rekistereihin, voi siihenkin liittyä valomerkkilaitteita, jotka sijaitsevat ohjaustaulussa.

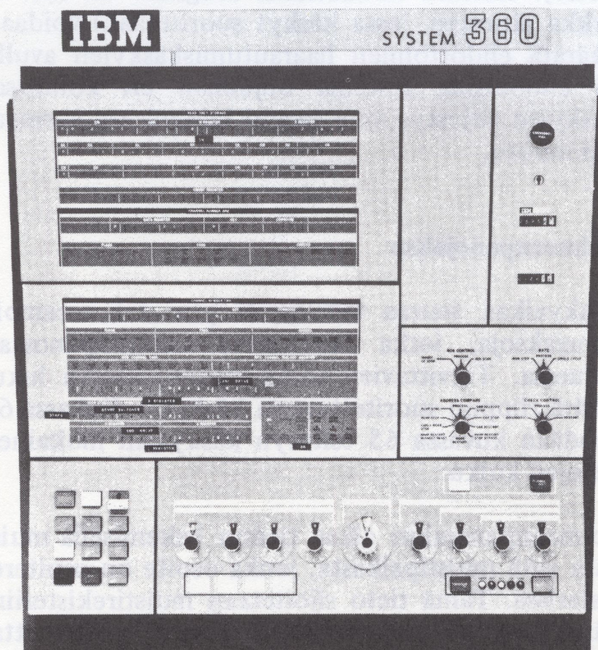
Yhteenlaskulaite

Yhteenlaskulaite saa tiedot kahdesta tai useammasta lähteestä, suorittaa yhteenlaskun ja lähettää tuloksen tulosrekisteriin. Kuvassa 64 on yhteenlaskuvirtapiirin kaksi positiota, joihin saadaan luvut rekistereistä A ja B. Yhteenlaskulaite muodostaa summan. Kussakin positiossa mahdollisesti ylimenevä osa (muistinumero) siirretään seuraavaan korkeimpaan positioon. Lopullinen summa sijoittuu tulosrekisterin vastaaviin positioihin.

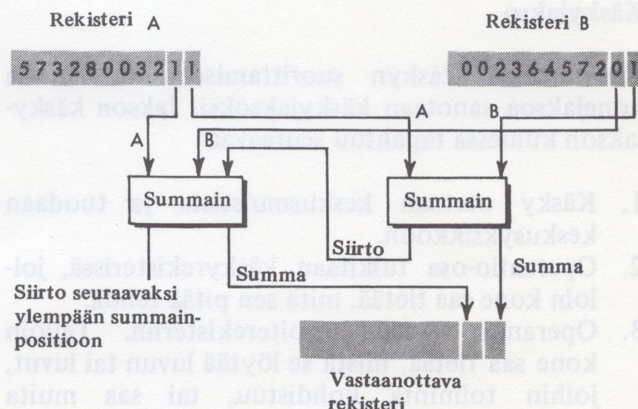
KONEJAKSOT

Kaikki konetoiminnat kestävät tietyn kiinteän pituisen aikavälin. Nämä aikavälit mitataan määrättyjen pulssien avulla, joita lähettää jaksoluvultaan aina 5 miljoonaan herziin nouseva elektroninen kello. Tietty kiinteä pulssien luku määrää jokaiseen peruskonekierrokseen kuluvan ajan.

Tietokoneiden suoritusarvoista puhuttaesssa käytetään aikayksikköinä millisekunteja, mikrosekunteja



Kuva 63. Ohjaustaulu



Kuva 64. Tietokoneen summaimet

ja nanosekunteja. Nämä termit eivät tunnu ehkä mielekkäiltä ennen kuin selviää esim. miten lyhyt aika on millisekunti. Esimerkkinä kelvannee vaikka 'silmänräpäys', joka ihmisestä tuntuu lyhyeltä, mutta tietokoneen kannalta katsoen vie kuitenkin verraten paljon aikaa - kymmenesosasekunnin eli 100 millisekuntia!

Seuraava taulukko sisältää tavallisimmat aikayksiköt ja niistä käytetyt lyhenteet:

0,1	= 1/10 sekuntia = 100 millisekuntia
0,001	= 1/1000 sekuntia = 1 millisekunti (ms)
0,000001	= 1/1000000 sekuntia = 1 mikrosekunti (μs)
0,000000001	= 1/1000000000 sekuntia = 1 nanosekunti (ns)

Konejakson aikana tietokone voi suorittaa määrätyn toiminnan. Yhden käskyn suorittamiseen tarvittavien toimintojen lukumäärä riippuu kyseessä olevasta käskystä. Eri konetoimintoja yhdistämällä saadaan suoritetuksi jokainen käsky.

Käskyissä on vähintään kaksi osaa: operaatio-osa ja operandi. Operaatio-osa ilmoittaa koneelle suoritettavan toiminnan: lukeminen, kirjoittaminen, yhteenlasku, vähennyslasku jne. Operandi voi olla tiedon tai käskyn osoite tai syöttö-tulostusyksikön tai jonkin muun laitteen osoite. Se voi myös eritellä sellaisia ohjaustoimintoja kuin suureen siirtäminen rekisterissä tai magneettinauhan takaisinkelaus.

Jotta keskusyksikkö voisi ottaa vastaan käskyn sekä tulkita ja suorittaa sen, sen täytyy toimia ennakolta määrätyssä järjestyksessä. Järjestys määräytyy kulloinkin kyseessä olevan käskyn mukaan ja toiminta suoritetaan kiinteässä koneen aikapulssien määräämässä ajassa.

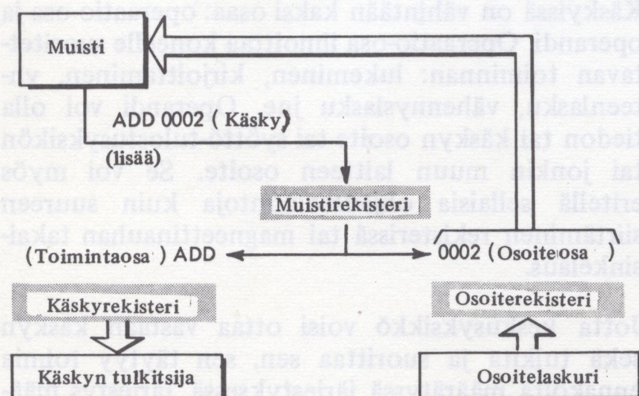
Käskyjakso

Ensimmäistä käskyn suorittamiseen tarvittavaa konejaksoa sanotaan käskyjaksoksi. Jakson käskyjakson kuluessa tapahtuu seuraavaa:

1. Käsky otetaan keskusmuistista ja tuodaan keskusyksikköön.
2. Operaatio-osa tulkitaan käskyrekisterissä, jolloin kone saa tietää, mitä sen pitää tehdä.
3. Operandi viedään osoiterekisteriin. Tällöin kone saa tietää, mistä se löytää luvun tai luvut, joihin toiminta kohdistuu, tai saa muita tarkempia ohjeita.
4. Määritetään seuraavan käskyn muistipaikka.

Ohjelman alussa on osoitelaskijaan asetettu ensimmäinen käskyn osoite. Tämä käsky haetaan muistista ja sitä suoritettaessa osoitelaskijaan automaattisesti asettuu järjestyksessä seuraavan käskyn osoite. Jos jokainen käsky vaatii yhden muistipaikan, laskija siirtyy yhden askeleen eteenpäin, jos käsky vaatii viisi muistipaikkaa, laskija siirtyy viisi askelta. Kun käsky on suoritettu, osoitelaskijaan on sijoitettu järjestyksessä seuraavan käskyn osoite. Laskijan siirtyminen tapahtuu automaattisesti. Toisin sanoen, jos tietokoneelle annetaan sarja käskyjä, se suorittaa ne yhden toisensa jälkeen, kunnes se saa käskyn toimia toisin.

Oletetaan, että on annettu käsky lisätä muistipaikan 2 sisältö laskulaitteen sisältöön. Kuva 65 näyttää tässä käskyssä kysymykseen tulevat rekisterit ja käskyn kulun.



Kuva 65. Käskyjakson toimintakaavio

Käskyjakso alkaa, kun osoitelaskija siirtää käskyn osoitteen osoiterekisteriin. Tämä käsky haetaan muistista ja tuodaan muistirekisteriin. Muistirekisteristä ohjataan operaatio-osa käskyrekisteriin ja operandi osoiterekisteriin. Operaatio-osan tulkitsemisen jälkeen kytkeytyvät vastaavat käskyn suorittamiseen tarvittavat virtapiirit.

Käskyjä ei tarvitse suorittaa ehdottomasti peräkkäin. Tietyt käskyt muuttavat käskyjen suoritusjärjestystä ilman ehtoja. Tässä tapauksessa muistista haettu käsky osoittaa, ettei suoritetakaan järjestyksessä seuraavana olevaa käskyä, vaan jossakin toisessa muistipaikassa oleva käsky. Osoitelaskijan normaali eteneminen muuttuu vastaavasti. Osoitelaskija voidaan asettaa esim. takaisin ohjelman alkuun niin, että sama ohjelma voidaan toistaa toiselle syötetylle tietoryhmälle.

Tämä haarautuminen toiseen käskyyn voi olla myös riippuvainen määrätyistä ehdoista. Tietokonetta voidaan ohjata tarkistamaan joitakin indikaattorilaitteita ja haarautumaan sen mukaan, onko indikaattori virittyneenä vai ei. Sellainen käsky on esimerkiksi seuraava: 'Tutki laskulaitteessa olevan suureen etumerkkiä, jos etumerkki on negatiivinen, hae seuraava käsky osoitteesta 5000, jos etumerkki on positiivinen, mene järjestyksessä seuraavaan muistipaikkaan'. Osoitelaskija asettuu vastaavasti osoittamaan toista kahdesta mahdollisesta muistipaikasta (5000 tai seuraavan käskyn osoite). Koneen noudattama looginen tie (so. se tarkka järjestys, jossa käskyt suoritetaan) voidaan määrätä ehdottomien haarautumiskäskyjen avulla tai ehdollisilla testeillä ohjelman eri kohdissa. Käskyjen järjestys muistissa ei kuitenkaan normaalisti muutu.

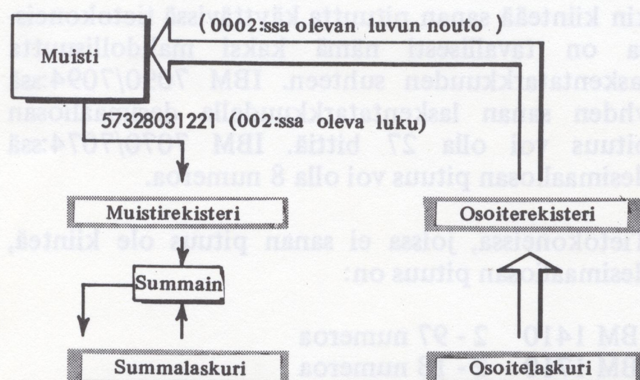
Toimeenpanojakso

Käskyaikaa seuraa tavallisesti yksi tai useampia konejaksoja, jotka tapahtuvat toimeenpanoajan kuluessa. Tarvittavien toimeenpanojaksojen lukumäärä riippuu suoritettavasta käskystä. Kuvassa 66 nähdään kuvassa 65 esitetyn käskyajan mukainen tietojen kulku.

Toimeenpanojakso alkaa tiedon hakemisella muistista siitä muistipaikasta, jonka osoite on osoiterekisterissä. Tämä tieto sijoitetaan muistirekisteriin. Tässä tapauksessa siirretään toinen yhteenlaskettavista laskulaitteesta otetun luvun kanssa yhteenlaskulaitteeseen. Muistirekisteristä ja laskulaitteesta

otetut arvot yhdistetään yhteenlaskulaitteessa ja summa palautetaan laskulaitteeseen.

Osoiterekisterissä voi olla myös muita tietoja kuin tietojen osoite. Se voi sisältää syöttö/tulostuslaitteen osoitteen tai suoritettavan ohjaustoiminnan. Käskyn operaatio-osa ilmoittaa koneelle, kuinka sen täytyy tulkita tämä tieto.



Kuva 66. Käskyjaksoa seuraavan toteutusjakson toimintakaavio

SARJA- JA RINNAKKAISTOIMINTA

Tietokoneiden sanotaan olevan joko sarja- tai rinnakkaistoimisia sen mukaan, miten kone suorittaa aritmeettiset laskutoimitukset. Olennaisesti kaikki aritmeettiset toiminnat suoritetaan yhteenlaskuna.

Sarjatoiminen kone suorittaa yhteenlaskun positio kerrallaan (ykköset, kymmenet, sadat jne.) aivan samalla tavalla kuin yhteenlasku suoritetaan kynällä ja paperilla. Jos summa jossakin positiossa nousee kaksinumeroiseksi, ylimenevä osa säilytetään ja lisätään seuraavaan korkeampaan positioon.

Sarjamenetelmällä tapahtuvaan toimintaan kuluva aika riippuu yhteenlaskettavien positioluvusta. Kuvassa 67 on esimerkki sarjamenettelystä yhteenlaskussa.

	1. vaihe	2. vaihe	3. vaihe	4. vaihe
Yhteen-laskettavat	1234 2459	1234 2459	1234 2459	1234 2459
siirto	1	1		
summa	3	93	693	3693

Kuva 67. Sarjayhteenlasku

Rinnakkaismenetelmää käyttävä tietokone lisää aina kokonaiset tietojaksot toisiinsa. Yhteenlasku, 'muistinumeroiden' liittäminen mukaanluettuna, vaatii aina saman positioluvusta riippumattoman ajan. Kuvassa 68 on esimerkki rinnakkaismenettelystä.

Yhteenlaskettavat	00564213 00000824
Siirto	1
Lopputulos	00565037

Kuva 68. Rinnakkaislasku

Kiinteä ja vaihteleva sananpituus

Käsitteet 'kiinteä' ja 'vaihteleva' sananpituus kuvaavat niitä tietoyksiköitä, joita tietokonejärjestelmä voi osoittaa ja käsitellä.

Kiinteätä sananpituutta käytettäessä kone käsittelee ja osoittaa tietoyksiköitä tai sanoja, joilla on kiinteä etukäteen määrätty positioluku. Sanan pituus on sovitettu järjestelmän rakenteen mukaan ja vastaa normaalisti pienintä tietoyksikköä, joka voidaan osoittaa käsiteltäväksi keskusyksikössä. Tietojaksot, kentät ja merkit eli tekijät käsitellään rinnakkaisesti sanoina. Rekisterit, laskijat, laskulaitteet ja muisti on konstruoitu sopiviksi käyttämään vakiopituista sanaa.

Vaihtelevaa sananpituutta käyttävässä työskentelytavassa on tietojen kulkua säätelevät virtapiirit kytketty siten, että tiedot käsitellään merkkeihin hajoitettuina. Tietojaksot ja kentät eli tekijät voivat olla mielivaltaisen pitkiä, kuitenkin muistikapasiteetin sallimissa rajoissa. Tieto ei ole käytettävissä sanoina vaan yksittäisinä merkkeinä.

Tietojenkäsittelyjärjestelmä voi työskennellä pelkästään kiinteällä sananpituudella, pelkästään vaihtelevalla sananpituudella tai käyttäen molempien menetelmien yhdistelmää.

TOIMINTA EKSPONENTTILUVUILLA

Matemaatikot ja tiedemiehet käyttävät logaritmeja yksinkertaistaakseen suurilla luvuilla pienillä desimaaliosilla suoritettavia laskutoimituksia. Samasta syystä ns. teknis-tieteellisissä tietokoneissa käytetään liukuvan pilkun aritmetiikkaa eli eksponenttiaritmetiikkaa (floating point). Osituskäyttöperiaat-

teella toimivien tietokoneiden ilmestyttyä markkinoille mitä moninaisimpia tehtäviä ja sovellutuksia varten, yleistyi eksponenttiaritmetiikkaa varten tarvittava lisälaite myös 'tavallisissa' tietokoneissa. Seuraava selostus koskee eksponenttioperaatioita IBM 7090/7094 tietokonejärjestelmissä. Samat periaatteet soveltuvat kuitenkin pieniä poikkeuksia lukuunottamatta myös ns. kaupallisiin tietokoneisiin sekä 'yleiskoneeseen' - Systeemiin/360.

Kaikissa tietokoneissa liukuvan pilkun aritmetiikka hoidetaan muuntamalla luvut ennen laskutoimituksia eksponenttimuotoon. Esimerkki:

N = b^e x f, missä
N = luku
b = lukujärjestelmän kanta (2 binäärijärjestelmässä, 10 desimaalijärjestelmässä jne.)
e = eksponentti (potenssi, johon kantaluku on korotettava, jotta lausekkeen arvoksi tulisi N.)
f = desimaaliosa.

Kantaluku riippuu tietokoneessa käytetystä sisäisestä lukujärjestelmästä. IBM 7090/7094:ssä kantaluku on 2 ja Systeemissä/360 se on 16.

Koska kantaluku on 'lyöty lukkoon' jo tietokonetta suunniteltaessa, se voidaan eliminoida edellisestä lausekkeesta, joka siten saa muodon:

N = eksponentti x mantissa

Eksponentin arvoalue määräytyy ensinnäkin lukuja varten varatun tilan perusteella ja toiseksi sen perusteella onko tilaa varattu eksponentin etumerkkiä (+ tai -) varten.
Eksponentin 'koko' on:

IBM 7090/7094:ssä kahdeksan bittiä ilman etumerkkiä eli 2^{127}.

Vastaavat desimaaliarvot ovat välillä 10^{38} - 10^{-38}

Systeemissä/360 seitsemän bittiä eli 16^{63}-16^{-64} ja vastaavat desimaaliarvot välillä 10^{76} - 10^{-77}.

Ellei eksponentille ole varattu merkkipostiota, eksponenttia 0 edustaa eksponentin arvoalueen keskipiste. Positiiviset eksponentit lisätään tähän arvoon ja negatiiviset taas vähennetään. Tätä 'skaalattua' eksponenttia nimitetään karakteristikaksi.

Tietokoneissa, joissa sanan pituus on kiinteä, desimaaliosan arvoalue määräytyy sen tilan mukaan mikä kokonaisen eksponenttiluvun esittämiseen varatusta sanasta (tai sanoista) jää jäljelle eksponentin ja sen etumerkin vaatiman tilan jälkeen. Systeemissä/360 desimaaliosassa voi olla korkeintaan 6 (tai 14) heksadesimaalinumeroa (24 tai 56 bittiä) riippuen siitä käytetäänkö yhden tai kahden 32 bitin sanan laskentatarkkuutta. Muissakin kiinteää sanan pituutta käyttävissä tietokoneissa on tavallisesti nämä kaksi mahdollisuutta laskentatarkkuuden suhteen. IBM 7090/7094:ssä yhden sanan laskentatarkkuudella desimaaliosan pituus voi olla 27 bittiä. IBM 7070/7074:ssä desimaaliosan pituus voi olla 8 numeroa.

Tietokoneissa, joissa ei sanan pituus ole kiinteä, desimaaliosan pituus on:

IBM 1410 2 - 97 numeroa
IBM 1710 2 - 18 numeroa

Nimitys 'liukuvan pilkun lukuesitys' johtuu siitä, että tällä merkintätavalla, eksponentti x desimaaliosa, voidaan eksponenttia vaihdella ja siten muuttaa desimaalipilkun paikkaa (taikka, koneesta riippuen, binääri- tai heksadesimaalipilkun paikkaa) luvun arvon silti muuttumatta (kuva 69).

Bittiarvot

Kokonaisosa | Murto-osa | ... sanan loppu

Bittiarvot	64	32	16	8	4	2	1.	1/2	1/4	1/8	1/16	1/32	1/64	1/128
37 = 1 X 37 = 2^0 - X		1	0	0	1	0	1.							
37 = 2 X 18-1/2 = 2^1 - X			1	0	0	1	0.	1						
37 = 4 X 9-1/4 = 2^2 - X				1	0	0	1.	0	1					
37 = 8 X 4-5/8 = 2^3 - X					1	0	0.	1	0	1				
37 = 16 X 2-5/16 = 2^4 - X						1	0.	0	1	0	1			
37 = 32 X 1-5/32 = 2^5 - X							1.	0	0	1	0	1		
37 = 64 X 37/64 = 2^6 - X							.	1	0	0	1	0	1	
37 = 128 X 37/128 = 2^7 - X							.	0	1	0	0	1	0	1
37 = 1/2 X 74 = 2^{-1} - X		1	0	0	1	0	1.							

Kuva 69. Eksponenttiaritmetiikan merkintätapoja

Kun desimaalipilkku sijoittuu välittömästi desimaaliosan korkeimman eli vasemmanpuoleisimman numeron vasemmalle puolelle, eksponenttiluku on normaalimuotoinen. Kun lukuja luetaan sisään tietokoneeseen eksponenttilukuina, ne muunnetaan juuri normaalimuotoon. Aritmeettisten operaatioiden yhteydessä pilkku saattaa siirtyä oikealle tai vasemmalle koska tietokone yrittää saada kaikki yhteen laskutoimitukseen kuuluvat luvut samaan 2:n pontenssiin (menettely on periaatteessa sama

kuin murtoluvuilla laskettaessa yhteisen nimittäjän etsiminen jotta laskutoimitus kävisi päinsä). Luvut eivät siis säily normaalimuotoisina. Lopputulos kaikissa laskutoimituksissa kuitenkin asetetaan normaalimuotoon.

Binäärinen eksponenttiesitys

Ennen varsinaisen aiheen käsittelyä muutama sana binäärikoneen toiminnasta eksponenttilukujen suhteen.

Karakteristikalla ei ole etumerkkiä. Tämän vuoksi eksponentti 0 sijoittuu karakteristikakentän (8 bitin) arvoalueen keskipisteeseen. Koska karakteristika ja desimaaliosa yleensä ilmoitetaan oktaalimuodossa, on kuvassa 70 esitetty suurimman, pienimmän ja 0-eksponentin desimaaliset arvot sekä vastaavat oktaali- ja binäärimuotoiset karakteristikat.

	Eksponentti	Desimaalinen karakteristika	Binäärinen karakteristika
$10^{38} \approx 2^{127}$	127	**255**	11 111 111
$10^{0} \approx 2^{0}$	0	128	10 000 000
$10^{-38} \approx 2^{-128}$	128	0	00 000 000

Kuva 70. Karakteristikan arvoalue (IBM 1130 tietokonejärjestelmä)

Binäärijärjestelmässä normaalimuotoisen eksponenttiluvun mantissan vasemmanpuoleisin bitti on aina 1. Desimaaliosa on sentähden aina yhtä suuri tai suurempi kuin 1/2, mutta pienempi kuin 1. Kuvassa 71 on yksinkertaisia esimerkkejä desimaalijärjestelmän lukujen muuntamisesta binäärisiksi, normaalimuotoisiksi eksponenttiluvuiksi.

Desim. luku	Desimaaliesitys	Eksponentti	Murto-osa	Karakteristika bittipositiot	Murto-osa bittipositiot
1	$= 2^{1} \times 1/2$	1	1/2	10 000 001	100 000 000
2	$= 2^{2} \times 1/2$	2	1/2	10 000 010	100 000 000
3	$= 2^{2} \times 3/4$	2	1/2+1/4	10 000 010	110 000 000
4	$= 2^{3} \times 1/2$	3	1/2	10 000 011	100 000 000
5	$= 2^{3} \times 5/8$	3	1/2+0/4+1/8	10 000 011	101 000 000
10	$= 2^{4} \times 5/8$	4	1/2+0/4+1/8	10 000 100	101 000 000
.5	$= 2^{0} \times 1/2$	0	1/2	10 000 000	100 000 000
.005	$= 2^{-2} \times 1/2$	-2	1/2	01 111 110	100 000 000

Kuva 71. Esimerkkejä normaalimuotoisista, binäärisistä eksponenttiluvuista

Eräs yksinkertainen menetelmä, jolla desimaalijärjestelmän kokonaislukuja voidaan muuntaa eksponenttimuotoon, on seuraava:

1. Luku muunnetaan oktaalimuotoon (ks. luku 'Oktaalijärjestelmä'). Esim. 37 on oktaalilukuna 45.
2. Oktaaliluku muunnetaan binääriseksi, esim. 45 = 100 101.
3. Binääripilkku (binäärijärjestelmän desimaalipilkku) sijoitetaan välittömästi luvun oikealle puolelle (100 101).
4. Binääripilkku siirretään luvun vasemmanpuoleisimman 1:n vasemmalle puolelle eli ensimmäisen merkitsevän numeron eteen (,100 101).
5. Luku on nyt desimaaliosa, jonka arvo on 1/2:n ja 1:n välillä.
6. Lasketaan monenko binääriposition verran pilkkua siirrettiin vasemmalle. Saatu luku on etsitty 2:n eksponentti (tässä tapauksessa 6).
7. Kohdasta 6 saatu eksponentti lisätään karakteristikan arvoalueen keskipisteeseen.

Edellisestä esimerkistä saadaan:

Karakteristika		Desimaaliosa
10000110		100 101
eli 2^{6} x 37/64	=	64 x 37/64 = 37

Desimaalijärjestelmän murtolukujen muunto eksponenttiluvuiksi on samankaltainen proseduuri paitsi, että binääripilkkua siirretään oikealle ja kasvatetaan 2:n negatiivista eksponenttia.

1. Luku muunnetaan oktaalimuotoon, esim. 0,149 = 0,1142 oktaalimuodossa.
2. Oktaaliluku muunnetaan binäärimuotoon (1142 = 001 001 100 010).
3. Binääripilkku sijoitetaan luvun eteen (0,001 001 100 010).
4. Pilkku siirretään ensimmäisen merkitsevän numeron (1:n) eteen (00,1 001 100 010).

5. Negatiivinen eksponentti on pilkkua siirrettäessä ohitettujen binääripositioiden lukumäärä (2).

6. Saatu luku vähennetään karakteristikan arvoalueen keskipisteestä (oktaaliarvo on 200 - 2 = 176).

Esim. karakteristika	desimaaliosa
01 111 110	100 110 001

$= 2^{-2} \times (1/2 + 1/16 + 1/32 + 1/152)$
$= 1/4 \times ((256 + 32 + 16 + 1)/512)$
$= 1/4 \times 305/512$
$= 305 / 2048$
$= 0{,}1489$ (n. 0,149)

Tietokone muokkaa eksponenttisanat esim. yhteenlaskua varten seuraavasti:

12 + 97

Desimaali-aritmetiikka	Oktaali-aritmetiikka
12	14
+ 97	+ 141
109_{10}	155_8

Binäärinen eksponenttiaritmetiikka

00,1100	eli	$2^4 \times 1100$
+00,1100001		$2^7 \times 1100001$

Normaalimuotoisina luvut ovat seuraavat:

Karakteristika (oktaali)	(binääri)	Desimaaliosa (binääri)
204	10 000 100	1100
207	10 000 111	1100001

Tietokone järjestää luvut uudelleen siten, että niiden karakteristikat ovat samat ja suorittaa yhteenlaskun sen jälkeen.

Karakteristika (oktaali)	(binääri)	Desimaaliosa (binääri)	
207	10 000 111	000 110 0	Karakteristikaan lisätään 3 ja mantissaan 3 etunollaa.
207	10 000 111	+110 000 1	Säilyy ennallaan
207	10 000 111	110 110 1	eli desimaalimuotoon muutettuna:

$$2^7 \times (1/2 + 1/4 + 1/16 + 1/32 + 1/128) = 109$$
$$128 \times [(64 + 32 + 8 + 4 + 1)/128] = 109$$

Koska tämä yksinkertainen yhteenlaskuesimerkki tuo esiin olennaisen eksponenttiaritmetiikkaan liittyvästä lukujen manipuloinnista tietokoneen sisällä, ei liene tarpeellista esittää muita laskutoimituksia.

Heksadesimaalinen merkintätapa

Systeemissä/360 samoin kuin binäärikoneissakin eksponentti 0 ilmaistaan karakteristikan arvoalueen keskipisteellä. Eksponentti on se 16:n potenssi, jolla heksadesimaalinen mantissa on kerrottava jotta tulokseksi saataisiin haluttu luku.

Desimaalisen luvun, esim. 149,25, muunto eksponenttimuotoon tapahtuu seuraavasti:

1. Luku hajoitetaan kokonais- ja desimaaliosiin: 149,25 = 149 + 0,25

2. Kokonaisosa muunnetaan heksadesimaalimuotoon (ks. kuva 31): $149_{10} = 95_{16}$

3. Desimaaliosa (mantissa) muunnetaan heksadesimaalimuotoon (kuva 32): $0{,}25_{10} = 0{,}4_{16}$

4. Osat yhdistetään ja muunnetaan normaalimuotoon (=desimaaliosa kerrottuna 16:n potenssilla): $95{,}4_{16} = (0{,}954 \times 16^2)_{16}$

5. Koska 64 on karakteristikan arvoalueen keskipiste, eksponentti (2) lisätään 64:ään. Tulos on oikea ('skaalattu') karakteristika: 64 + 2 = 66 1000010

6. Desimaaliosa muunnetaan binäärimuotoon: $0{,}954_{16} = 0{,}1001\ 0101\ 0100$

7. Siten luku 149,25 (heksadesimaalisena 95,4) voidaan merkitä eksponenttimuodossa (yhtenä sanana eli lyhyellä laskentatarkkuudella) seuraavasti:

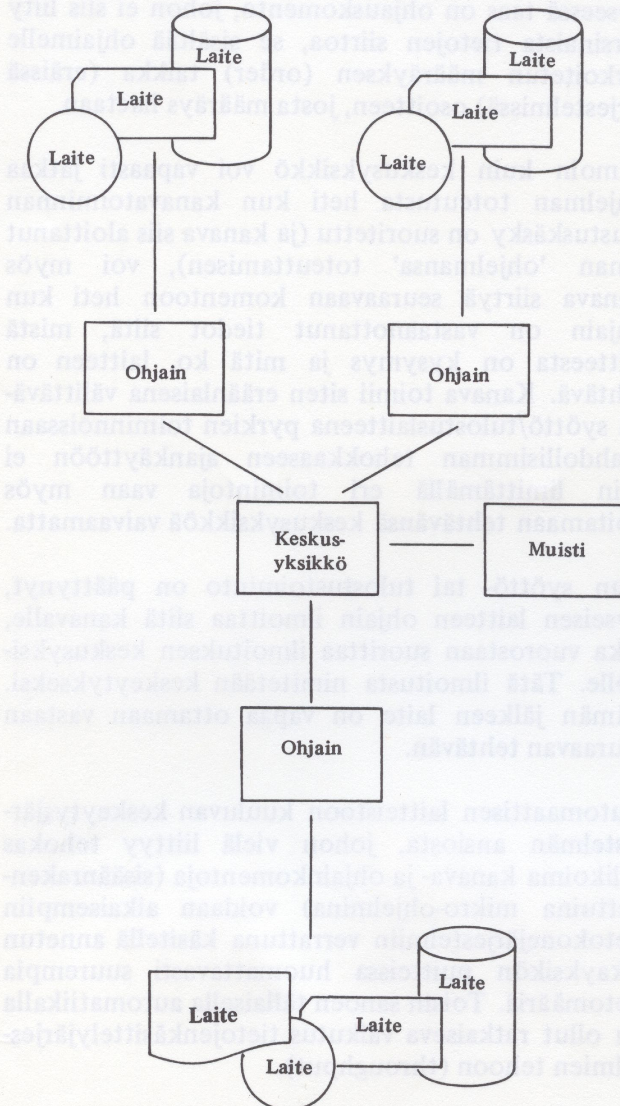
Etumerkki*)	Karakteristika
0	100 0010

Desimaaliosa
1001 0101 0100 0000 0000 0000

*) Etumerkki on 0, koska alkuperäinen luku on positiivinen.

SYÖTTÖ- JA TULOSTUSLAITTEET

Syöttö-ja tulostusyksiköt ovat laitteita, joista tietoja siirretään keskusmuistiin ja joihin tietoja siirretään keskusmuistista (kuva 72).



Kuva 72. Syöttö- ja tulostuslaitteet tietojenkäsittelyjärjestelmässä

Yleensä laitteen toiminnan panee alulle ohjelmassa annettu käsky, jonka perusteella kehitetään I/O-kanavalle toimintakomento. Kanavaan kytketty ohjain tunnistaa ja selvittää komennon sekä toteuttaa halutun toiminnon.

Jotkut ohjaimet pystyvät hoitamaan vain tietyn tyyppisiä laitteita, esim. magneettinauhayksiköitä. Toiset taas voivat valvoa useita eri laitteita, esim. IBM 2841 ohjaimeen voidaan liittää poimintamuistilaitteet 2302, 2303, 2311 ja 2321.

Syöttölaitteet pystyvät lukemaan tai tunnistamaan tietoja reikäkorteilta, magneettinauhalta, paperinauhalta, lomakkeilta magneettisesti tai optisesti, mikrofilmiltä jne. Tietokoneesta etäällä sijaitsevat päätteet voivat myös toimia syöttö- ja tulostuslaitteina tietoliikenneyhteyksien avulla. Luettu tieto siirtyy systeemin keskusmuistiin käsittelyä varten. Tulostuslaitteet voivat tallentaa tai rekisteröidä tietoja keskusmuistista reikäkorteille, magneettinauhalle, paperinauhalle, lomakkeille, mikrofilmille, näyttölaitteen kuvaputkelle jne.

Lukeminen tapahtuu siten, että syöttömateriaali liikkuu fyysisesti syöttölaitteen läpi. Tieto luetaan ja muunnetaan tietokoneelle sopivaan muotoon, siirretään keskusmuistiin.

Kirjoitukseen kuuluu keskusmuistissa olevien tietojen muuntaminen tulostusmateriaalille sopivaan muotoon tai kielelle ja tiedon tallennus tulostusmateriaalille tulostuslaitetta käyttäen.

Useimmat syöttö/tulostuslaitteet ovat automaattisia: kun ne on kerran käynnistetty, ne jatkavat toimintaansa muistiin tallennetun ohjelman ohjaamina. Tarvittava laite ilmoitetaan käskyssä, joka panee laitteen lukemaan tai kirjoittamaan. Käskyssä ilmoitetaan myös osoite, johon tieto sijoitetaan tai josta se otetaan.

Muutamat harvat syöttölaitteet ovat käsinohjattavia ja sellaisista laitteista puuttuu aina tiedon tallennusmateriaali. Tiedot sijoitetaan aina sen sijaan suoraan muistiin näppäimillä tai kytkimillä, jotka tavallisesti sijaitsevat tarkkailupöydässä.

Tällaisia, suoraan tietokoneeseen liitettyjä laitteita voivat olla näppäimistöllä varustettu ohjauspöytä, pääte (esim. IBM 1050) tai näyttölaite. Lisäksi voidaan käyttää monenlaisia tietojen kaukokäsittelylaitteistoja (joiden ei tarvitse sijaita tietokoneen välittömässä läheisyydessä). Monissa tapauksissa näihin kaukokäsittelylaitteistoihin tai päätteihin kuuluu tietojen tallennusvälineen sijasta muisti, johon esim. näppäimistön avulla kirjoitettu tieto tallentuu siksi kunnes ko. pääte saa keskusyksiköstä pyynnön siirtää tieto keskusmuistiin.

OHJAIMET

Eräs ohjaimen tehtävistä on puskurointi, jolla tarkoitetaan syöttö- ja tulostuslaitteiden sekä keskusyksikön toimintojen koordinointia. Muita yleisiä tehtäviä ovat tietojen tarkistus, koodaus ja koodien tulkinta. Jos ohjaimeen on liitetty useita samanlaisia laitteita, sen päätehtäviin kuuluu lisäksi 1) laitteiden prioriteetin määritys toimintojen suhteen ja 2) laiteidentifikaation ilmoittaminen (keskusyksikölle) kun kyseessä on syöttölaite. Tulostuksessa taas ohjaimen tehtävä on ohjata tiedot haluttuun tulostusyksikköön.

Joissakin tietojenkäsittelyjärjestelmissä käytetään tietojen ohjaustoiminnan käskyistä termiä 'määräys' (order). Ohjelmaan sisältyvä määräys koostuu tällöin vain sen laitteen osoitteesta, jota seuraava luku- tai kirjoitustoiminto koskee. Systeemissä/360 määräykset muodostavat osan ohjauskomennoista ja ne annetaan kanavan kautta ohjaimelle: määräyksen spesifioima toiminto, aputoiminto, johon ei liity varsinaista tietojen siirtoa (nauhan kelaus, haku tms.) koskee kuitenkin syöttö- tai tulostuslaitetta - ohjain huolehtii sen toteutuksesta.

KANAVAT

Ohjain voi olla osa syöttö/tulostuslaitetta tai laitteiden läheisyydessä sijaitseva erillinen 'laatikko'. Samoin kanava (tai joukko kanavia) voi sisältyä keskusyksikköön tai olla erillinen, keskusyksikön läheisyydessä sijaitseva laite. Kanavaa voidaan pitää eräänlaisena ohjaimena, jonka tehtävänä on hoitaa syöttö- ja tulostuslaitteiden ohjaimia. Itse asiassa kanava on pieni erillinen 'tietokone', jonka yksinomaisena tehtävänä on valvoa ja ohjata syöttö- ja tulostuslaitteita ja näiden ohjaimia. Kun kanava on 'aktivoitu' erityisen alustuskäskyn toteutuksen jälkeen, se suorittaa yhden tai useamman komennon määrittämät toiminnot samaan tapaan kuin ohjelmassa tapahtuu esim. alirutiinin toteutus. Erityisesti on huomattava, että kanavakomentojen toteutus tapahtuu limitettynä keskusyksikön toiminnan kanssa. Tämä merkitsee sitä, että ohjelman toteutus jatkuu samanaikaisesti kun kanavalla tapahtuu tietojen siirtoa keskusmuistin ja syöttö/tulostuslaitteiden välillä.

Joissakin tapauksissa tapahtuu myös kanavatoimintojen limitys eli useiden syöttö- ja tulostuslaitteiden samanaikainen toiminta.

Ohjelman toiminta-alkioita nimitetään käskyiksi (instructions). Kanavan käskyistä käytetään nimitystä komento (command). Komento sisältää toimintakoodin (esim. lue, kirjoita tms.) kuten muutkin käskyt. Jos komentoon liittyy tietojen siirto, se sisältää myös osoitteen, joka määrittää mistä tieto kirjoitetaan tai mihin se luetaan, jos kyseessä taas on ohjauskomento, johon ei siis liity varsinaista tietojen siirtoa, se sisältää ohjaimelle tarkoitetun määräyksen (order) taikka (eräissä järjestelmissä) osoitteen, josta määräys haetaan.

Samoin kuin keskusyksikkö voi vapaasti jatkaa ohjelman toteutusta heti kun kanavatoiminnan alustuskäsky on suoritettu (ja kanava siis aloittanut oman 'ohjelmansa' toteuttamisen), voi myös kanava siirtyä seuraavaan komentoon heti kun ohjain on vastaanottanut tiedot siitä, mistä laitteesta on kysymys ja mitä ko. laitteen on tehtävä. Kanava toimii siten eräänlaisena välittävänä syöttö/tulostuslaitteena pyrkien toiminnoissaan mahdollisimman tehokkaaseen ajankäyttöön ei vain limittämällä eri toimintoja vaan myös hoitamaan tehtävänsä keskusyksikköä vaivaamatta.

Kun syöttö- tai tulostustoiminto on päättynyt, kyseisen laitteen ohjain ilmoittaa siitä kanavalle, joka vuorostaan suorittaa ilmoituksen keskusyksikölle. Tätä ilmoitusta nimitetään keskeytykseksi. Tämän jälkeen laite on vapaa ottamaan vastaan seuraavan tehtävän.

Automaattisen laitteistoon kuuluvan keskeytysjärjestelmän ansiosta, johon vielä liittyy tehokas valikoima kanava- ja ohjainkomentoja (sisäänrakennettuina mikro-ohjelmina) voidaan aikaisempiin tietokonejärjestelmiin verrattuna käsitellä annetun aikayksikön puitteissa huomattavasti suurempia tietomääriä. Toisin sanoen tällaisella automatiikalla on ollut ratkaiseva vaikutus tietojenkäsittelyjärjestelmien tehoon (throughput).

KELPOISUUSTARKISTUKSET

Kaikkien syöttö/tulostusyksiköiden ja muistin välillä siirrettyjen tietojen kelpoisuus tarkistetaan kahdella tavalla. Ensiksi, syöttölaite tarkistaa tiedot ennen kuin se lähettää ne keskusyksikköön ja myös tulostuslaite tarkistaa tiedot, kun se ottaa ne vastaan. Toiseksi, varsinainen tietojen tarkistus tapahtuu keskusyksikössä, kun se vastaanottaa tietoja tai lähettää niitä. Näissä tarkistuksissa ei huomata väärien tietojen käyttöä. Jos esimerkiksi

4:n tilalla on 5, kone ei voi tätä virhettä huomata. Mutta jos numero tai merkki on materiaalilla tai koneessa väärin esitettynä tai koodattuna, tämä virhe huomataan automaattisesti.

ILMAISIMET, NÄPPÄIMET JA KYTKIMET

Kaikissa syöttö/tulostuslaitteissa on indikaattorivaloja ja toimintanäppäimiä ja -kytkimiä (kuva 73). Ilmaisinvalot näyttävät yksikön tilan: kytketty tai ei, valmis käyttöön, käytössä jne. Toimintanäppäimiä ja -kytkimiä käytetään etupäässä käsin tapahtuvaan toimintojen aloittamiseen ja päättämiseen. Ilmaisimien, näppäinten ja kytkimien erikoistehtävät ja käyttö selvitetään yksityisten koneiden ja järjestelmien käsikirjoissa.

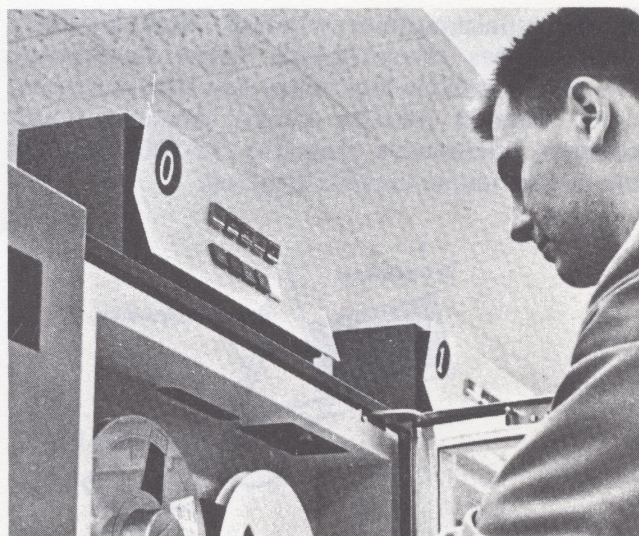
KYTKENTÄLAATTA

Jotkut syöttö/tulostuslaitteet on varustettu kytkentätaululla. Taulun avulla voidaan muotoilla, järjestellä, mitätöidä ja valikoida laitteen läpi kulkevia tietoja.

Kytkentätaulu on periaatteessa samanlainen kuin puhelinkeskuksen kytkentäpöytä. Tuleva puhelu sytyttää merkkilampun, joka kertoo keskuksen hoitajalle, mitä linjaa pitkin puhelu on tulossa. Vastattuaan soittoon keskuksen hoitaja työntää johtimen päässä olevan pistotulpan koskettimeen, joka on pöydän sisällä kytketty haluttuun sivupuhelimeen. Todellisuudessa keskuksen hoitaja muodostaa tällöin sähköisen virtapiirin halutun tuloksen aikaansaamiseksi.

Koneen kytkentätaulu muodostaa sisäisiä virtapiirejä tauluun sijoitettujen johtimien avulla. Varsinainen kytkeminen tapahtuu otto- ja antojakeiksi kutsuttujen reikien läpi. Antojakit lähettävät pulsseja ja ottojakit vastaanottavat niitä. Käytettävät anto- ja ottojakit riippuvat suoritettavista tehtävistä. Operaattori sijoittaa johdot tauluun eli kytkee sen jokaista erikoistehtävää varten ennen kuin taulu sijoitetaan koneeseen.

Kytkentätauluilla varustetut syöttöyksiköt voivat muuttaa tai vaihtaa tietoja sen jälkeen kun yksikkö on ne lukenut, mutta ennen kuin ne on lähetetty järjestelmän keskusmuistiin. Tulostusyksiköt voivat muuttaa tai vaihtaa tietoja sen jälkeen kun ne ovat tulleet keskusmuistista, mutta ennen kuin tiedot on ehditty lävistää tai kirjoittaa.



Kuva 73. Toimintanäppäimistö

Kytkentätaulut ovat irroitettavia, joten niitä voidaan helposti vaihtaa käyttötarkoitusten mukaisesti tai jotakin erillistä taulua muutetaan siten, että se soveltuu käytettäväksi kaikissa toiminnoissa.

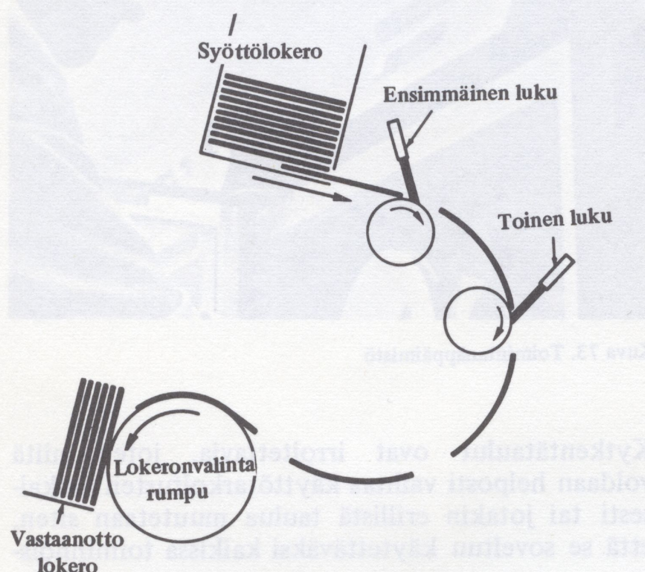
KORTINLUKIJAT

Kortteja lukevat laitteet siirtävät reikäkorteille lävistetyt tiedot tietokoneeseen. Lukija siirtää kortit lukuyksikön läpi, joka muuntaa kortilla olevat tiedot elektroniseen muotoon. Lukuyksiköitä on käytössä kahdenlaisia: lukuharjoilla tai valokennoilla varustettuja.

Lukuharjoilla varustetussa lukuyksikössä kortit siirtyvät syöttölokerosta syöttölaitteen läpi lukevien harjojen alitse. Lukuharjat tunnistavat sähköisesti kortin jokaisessa sarakkeessa olevat lävistykset tai niiden puuttumisen (kuva 74). Tämä sähköinen tunnistaminen muuttaa kortilla olevan tiedon sähköimpulsseiksi, joita lukulaitteen virtapiirit voivat käyttää hyväkseen ja tallentaa tietona muistiin. Kun kortit on luettu, ne poistetaan syöttöradalta ja sijoitetaan vastaanottolokeroon samaan järjestykseen kuin missä ne syötettiinkin. Joissakin lukijoissa on kaksi lukuasemaa, niin että jokainen kortti voidaan lukea kahdesti sen kulkiessa syöttöradan läpi lukemistoiminnan virheettömyyden tarkistamiseksi.

Valokennoilla varustettu lukija suorittaa samat toimenpiteet kuin harjoilla varustettukin: ero on

reikien tunnistamismenetelmässä. Kun lävistetty kortti ohittaa lukuyksikössä olevan valolähteen, lävistettyjen reikien läpi tuleva valo aktivoi kennot, yhden kennon jokaista kortin saraketta kohti. Korttien lukunopeus vaihtelee 12-1000 korttiin minuutissa lukijan mallista riippuen.

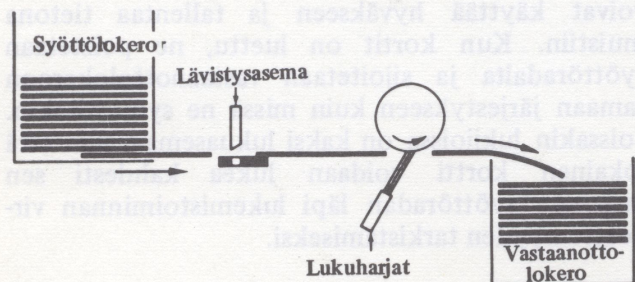


Kuva 74. Kortin syöttö ja luku

KORTINLÄVISTIMET

Tietokonejärjestelmän reikäkorttitulostus hoidetaan kortinlävistimen avulla. Lävistin siirtää automaattisesti lävistämättömät kortit syöttölokerosta yhden kerrallaan lävistysmekanismin alle, joka lävistää muistista siirrettyä tietoa (kuva 75). Lävistämisen jälkeen kortti siirtyy tarkistusasemalle, jossa juuri lävistetty tieto luetaan ja tarkistetaan vertaamalla sitä lävistysasemalle siirrettyyn tietoon. Sen jälkeen kortti siirtyy vastaanottolokeroon.

Lävistysnopeus vaihtelee 12-500 korttiin minuutissa riippuen käytetystä lävistimestä.



Kuva 75. Lävistettävän kortin syöttö

MAGNEETTINAUHAYKSIKÖT

Tietokoneiden sisäisen nopeuden kasvu vaatii nopeita syöttö- ja tulostuslaitteita, jotta systeemin toiminnassa vältyttäisiin turhilta tietojen syöttöön ja tulostukseen liittyviltä viiveiltä. Magneettinauhayksiköiden, joita voidaan käyttää sekä syöttö- että tulostuslaitteina, kehitys on tässä suhteessa erinomainen esimerkki: tietojen siirtonopeus on jatkuvasti kasvanut samoin kuin nauhakelan kapasiteetikin.

Vaikka kaikki magneettinauhayksiköt toiminnaltaan ovat pääpiirteiltään samanlaisia, erinäiset rakenteelliset parannukset ovat aiheuttaneet eroavuuksia, lisänneet nauhojen käyttökelpoisuutta mitä erilaisimpiin sovellutuksiin sekä helpottaneet niiden käyttöä esim. operaattorin kannalta.

IBM 2400-sarjan magneettinauhayksiköissä nauhan liike lukupään kautta on jatkuvaa ja sen nopeus on vakio. Nauha on siis aina luku- tai kirjoitustoiminnan aikana liikkeessä. Nopeus on 37,5, 75 tai 112,5 tai 200 tuumaa sekunnissa nauhayksiköstä riippuen.

Täydessä kelassa on nauhan pituus 2400 jalkaa (1/2" nauhaa), nauhakelan paino on alle 2 kiloa ja sille voidaan tallentaa noin 400 000 täyteen lävistettyä reikäkorttia vastaava tietomäärä.

Nauhayksikön saattaminen toimintakuntoon

Nauhayksikkö täytyy saattaa toimintakuntoon, ennen kuin se voi lukea tai tallentaa tietoa nauhalle. Näihin valmisteluihin kuuluu kahden nauhakelan asettaminen yksikköön ja nauhan pujottaminen nauhaa liikuttavaan mekanismiin. Pujottamisen helpottamiseksi luku-kirjoituspää voidaan irroittaa ja sulkea sen jälkeen uudestaan, jotta luku-kirjoitustoiminta voi tapahtua. Nauhan pujotustoiminta vastaa projektorin nauhan asetusta (kuva 78).

Yksikön toiminnan aikana nauha siirtyy rekisterikelalta vasemman tyhjiökanavan kautta luku-kirjoituspään toiselle puolelle ja oikean tyhjiökanavan kautta konekelalle. Nauha kulkee tyhjiökanavien kautta, ettei se katkeaisi hyvin nopeissa liikkeellelähdöissä ja pysäytyksissä. Pystysuoria tyhjiökanavia käytetään esim. 2400-sarjan nauhayksiköissä. Vaakasuoria kanavia käytetään vaihtelevan syöttönopeuden omaavissa koneissa.

Molemmat nauhakelat voivat pyöriä toisistaan riippumattomasti tyhjiökanavissa olevien kytkimien ohjaamina. Rekisterikela syöttää nauhaa, kun vasemmassa tyhjiökanavassa oleva nauhasilmukka saavuttaa minimipituutensa, ja konekela kelaa nauhaa, kun oikeassa tyhjiökanavassa oleva nauha saavuttaa lähellä kanavan pohjaa olevan pisteen.

Nauha voidaan kelata takaisin kelan alkuun joko kokonaan tai vähän kerrallaan takaisin päin. Takaisinkelausnopeus on noin 500 tuumaa sekunnissa.

Kasettien asetus

IBM 2420-yksikkö on automaattisesti toimintakunnossa sen jälkeen kun operaattori on asettanut nauhakelan yksikköön. Tässä yksikössä ei siis tarvitse suorittaa mitään aikaa vieviä käsin tapahtuvia pujotustoimintoja (kuva 79).

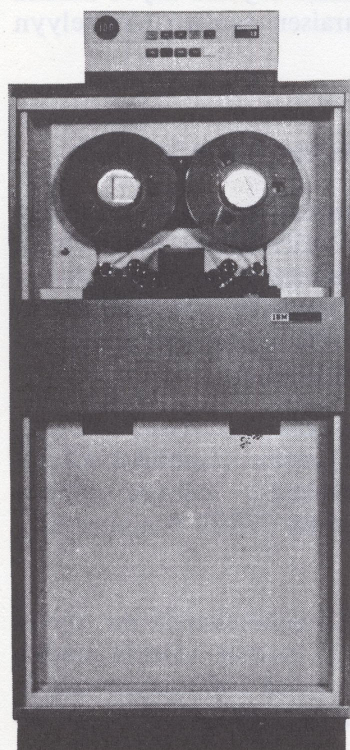
Magneettinauhan luku ja kirjoitus

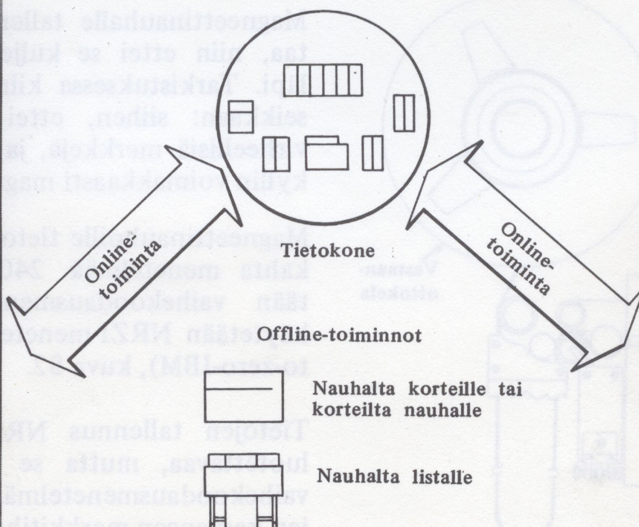
Magneettinauhayksikkö lukee ja tallentaa tietoja nauhan siirtyessä luku-kirjoituspään ohi. Nykyisissä magneettinauhayksiköissä käytetään kahdenlaisia luku-kirjoituspäitä, mutta molemmat tyypit suorittavat luku-kirjoitustoiminnan samojen periaatteiden mukaan (kuva 80).

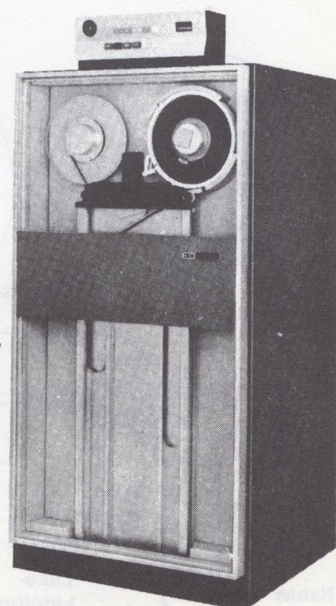
Kun magneettinauhalle tallennetaan tietoa, aikaisemmin tallennettu tieto tuhoutuu uuden tiedon alta. Luettaessa tieto ei tuhoudu, vaan sama tieto voidaan lukea niin monta kertaa kuin halutaan.

Tieto tallennetaan nauhalle magnetisoituina alueina nauhan pituussuunnassa oleviin uriin.

Kirjoituspäässä on jokaista tallennettavaa uraa varten yksi kirjoituskäämi, joiden kautta kulkeva sähkövirta magnetisoi liikkuvan nauhan rautaoksidipäällysteen ja tuhoaa aikaisemmin kirjoitetun tiedon (kuva 81).



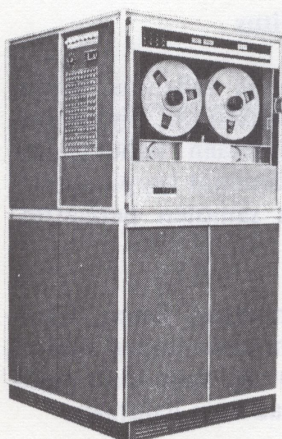
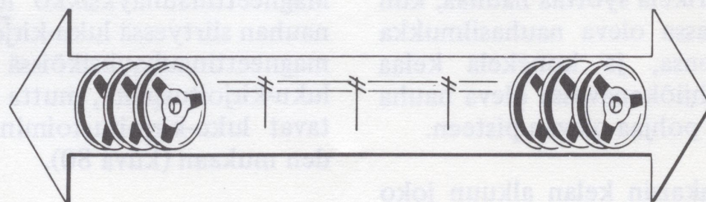
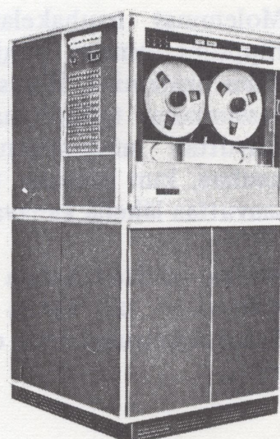




IBM 2401 nauhayksiköihin on nauha asetettava käsin. Nauhayksiköt soveltuvat sekä 'on-line'- että 'off-line'-toimintaan.

IBM 2420 malli 7 nauhayksiköissä käytetään nauhakasetteja, joilla nauhan pujotus ja takaisinkelaus tapahtuvat automaattisesti.

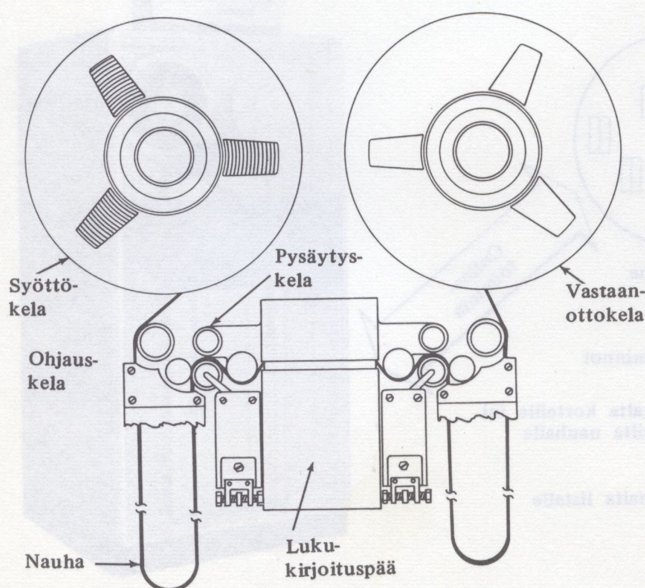
Kuva 76. IBM 2401 Magneettinauhayksiköitä

IBM 7702 magneettinauhapäätteiden avulla tiedon siirto tapahtuu nauhalta nauhalle.

Kuva 77. IBM Magneettinauhapäätteitä

Incremental-nauhayksiköissä nauha ei liiku tallennuksen tapahtuessa, mutta tallennettujen bittien koko on suunnilleen sama kuin muissakin nauhayksiköissä. Vaihtelevan nauhanopeuden omaavissa koneissa tallennettu tieto voidaan käyttää muissa yksiköissä ja päinvastoin.



Luku-kirjoitustoiminnassa magneettinauha siirtyy vaihdettavalta tiedostokelalta luku-kirjoituspään kautta nauhayksikön kiinteälle vastaanottokelalle. Nauhan peruutuksessa tai takaisinkelauksessa liike on päinvastainen.

Kuva 78. Magneettinauhan syöttömekanismi kaaviokuvana

Pysäytysperiaatteella (incremental) ja vaihtelevalla nopeudella toimivissa magneettinauhayksiköissä kirjoitettua nauhaa voidaan käyttää myös muissa yksiköissä, joissa on 7-uraisen nauhan käsittelyyn tarkoitettu lisälaite.

Magneettinauhalle tallennetun tiedon tarkistus

Magneettinauhalle tallennettu tieto täytyy tarkistaa, niin ettei se kulje virheellisenä järjestelmän läpi. Tarkistuksessa kiinnitetään huomio kahteen seikkaan: siihen, ettei nauhalle ole tallennettu virheellisiä merkkejä, ja että tallennetut bitit ovat kyllin voimakkaasti magnetoituja.

Magneettinauhoille tietoa tallennettaessa käytetään kahta menetelmää. 2400-sarjan yksiköissä käytetään vaihekoodausmenetelmää, kaikissa muissa käytetään NRZI-menetelmää (NRZI = non-return-to-zero-IBM), kuva 82.

Tietojen tallennus NRZI-menetelmällä on hyvin luotettavaa, mutta se on saanut väistyä uuden vaihekoodausmenetelmän tieltä 2400-sarjan nauhojen kasvaneen merkkitiheyden vuoksi.

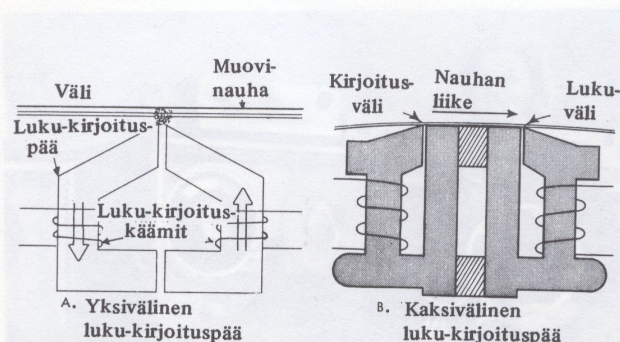
NRZI-nauhoilla käytetään virheenetsintämenetelmänä pariteettitarkistusta. Näin voidaan saada selville lähes kaikki luku- ja kirjoitusvirheet (kuva 83).

Pariteettitarkistus paljastaa virheen, mutta ei sen sijaan pysty määrittämään virheen laatua. Kahdessa



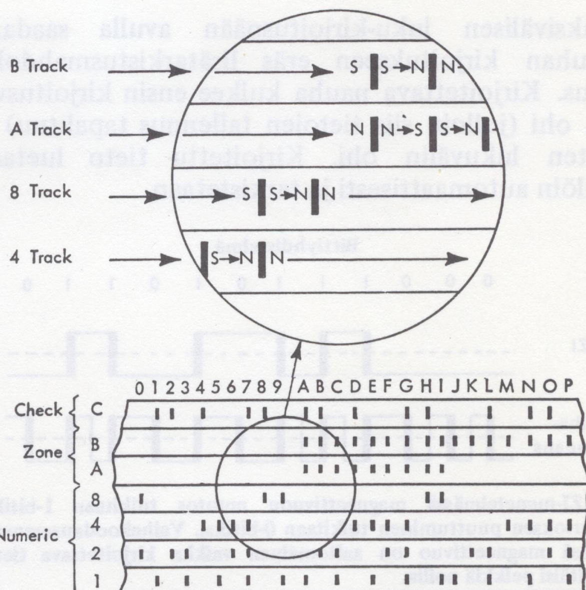
Kuva 79. Nauhakasetin asetus

merkissä samanaikaisesti esiintyvät samanlaiset kahden bitin virheet voisivat tietenkin toisensa kumoamalla antaa tulokseksi oikean pariteetin ja jäädä siten havaitsematta, mutta tällaiset yhteensattumat ovat äärimmäisen harvinaisia.



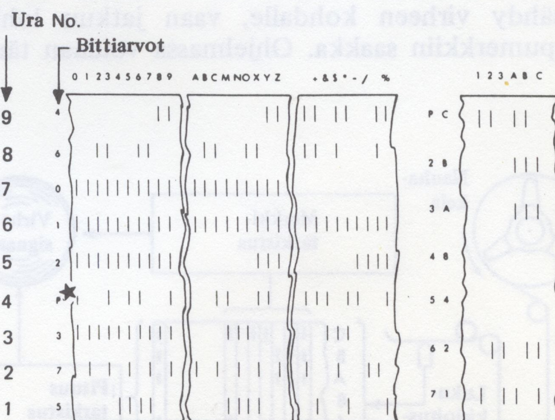
Yksivälisellä luku-kirjoituspäällä toiminta tapahtuu aina samalla välillä. Kaksivälisissä luku-kirjoituspäissä toiminta jakautuu siten, että ensimmäistä väliä käytetään kirjoitukseen ja toista lukuun, tällä järjestelyllä päästään kirjoitetun tiedon kelpoisuus toteamaan välittömästi.

Kuva 80. Magneettinauhayksikön luku-kirjoituspäät



Uuden tiedon rekisteröinti magneettinauhalle tapahtuu muuttamalla sähkövirran suuntaa ja tiettyjen kirjoituskäämien polariteettia. Tämä muutos vaikuttaa samansuuntaisesti vastaaviin nauhan uriin. Nauhalla poikittaissuunnassa olevat 0- ja 1-bitit edustavat keskusyksiköstä saapunutta tietoa.

A. Tiedon rekisteröinti 7-uraiselle magneettinauhalle BCD-koodilla.
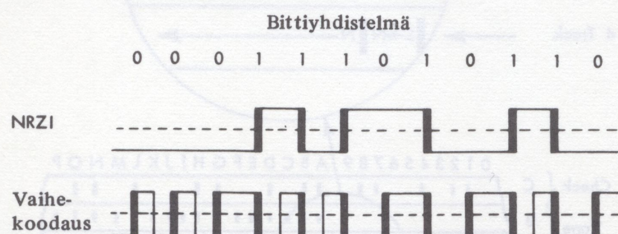


★ P-bitillä varmistetaan se, että bittien lukumäärä per merkki tai tavu on pariton.

B. Tiedon esitys 9-uraisella (EBCDIC) ja 7-uraisella magneettinauhalla.

Kuva 81. Tietojen rekisteröinti magneettinauhalle

Yhden uran virheet korjataan 2420:n mallissa 7 ajon aikana nauhan liikkuessa ilman, että vähennetään koneen suorituskykyä. Tästä johtuen useimmat korjaukset tehdäänkin käsittelyä keskeyttämättä.

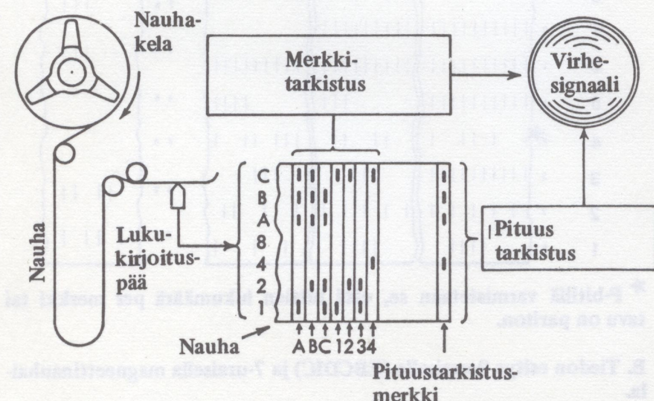
Kaksivälisen luku-kirjoituspään avulla saadaan nauhan kirjoitukseen eräs lisätarkistusmahdollisuus. Kirjoitettava nauha kulkee ensin kirjoitusvälin ohi (jolloin siis tietojen tallennus tapahtuu) ja sitten lukuvälin ohi. Kirjoitettu tieto luetaan tällöin automaattisesti ja tarkistetaan.



NRZI-menetelmässä magneettivuon muutos tulkitaan 1-bitiksi, muutoksen puuttuminen tulkitaan 0-bitiksi. Vaihekoodausmenetelmässä magneettivuo on aaltomainen vaikka kirjoitettava tietue sisältäisi pelkkiä nollia.

Kuva 82. NRZI- ja vaihekoodausmenetelmien vertailu

Kun kirjoitustoiminnassa tapahtuu virhe, se havaitaan välittömästi. Indikointi virheestä on myös keskusyksikössä olevan ohjelman käytettävissä ja ohjelmasta myös riippuu mitä toimenpiteitä virheen johdosta suoritetaan. Nauhan liike ei pysähdy virheen kohdalle, vaan jatkuu lohkon loppumerkkiin saakka. Ohjelmassa voidaan tämän



Nauhalta luettava tieto tarkistetaan kahdella tavalla. Merkkitarkistus (vertikaalinen) varmistaa, että kussakin merkissä on bittien lukumäärä parillinen. Jos kohdataan merkki, jossa bittien määrä on pariton, se saattaa merkitä virhettä (riippuen siitä, minkälaista pariteettia kulloinkin käytetään). Lohkokohtainen pituustarkistus (longitudinal check) suoritetaan siten, että lasketaan bittien lukumäärä per magneettinauhan ura (tarkistusbitit mukaanluettuina). Tässäkin tapauksessa bittien määrän on oltava parillinen, parittomuus merkitsee virhettä.

Kuva 83. 7-uraisen nauhan kelpoisuusmerkkitarkistukset, BCD-koodi, bittimäärä (pariteetti) parillinen

jälkeen antaa nauhayksikölle takaisinkelauskäsky sekä uudelleenkirjoituskäsky. Myös tähän uusintayritykseen sisältyvät kaikki virhetarkistukset.

Tietueet, jaksoväli ja nauhamerkki

Nauharekordeilla ei ole yleensä muita pituusrajoituksia kuin muistikapasiteetti. Ne voivat sisältää lähes mielivaltaisen määrän merkkejä, sanoja, kenttiä tms. Käytännön sovellutukset saattavat tietenkin asettaa lohkojen ja tietueiden pituuksille omat rajoituksensa.

Nauhalla olevia tietolohkoja (jotka voivat sisältää yhden tai useampia tietueita) erottaa toisistaan lohkoväli eli tyhjä väli, joka pituudeltaan on 2400-sarjan nauhoilla 0,6 tuumaa ja muilla nauhoilla 0,75 tuumaa. Nauhan kirjoituksessa tällainen tyhjä väli luodaan automaattisesti jokaisen lohkon perään. Nauhan luku taas alkaa tyhjää väliä seuraavasta ensimmäisestä merkistä ja jatkuu seuraavaan väliin asti. Lohkovälin avulla hoidetaan myös nauhan pysähtymiseen ja liikkeellelähtöön liittyvät ajoituskysymykset.

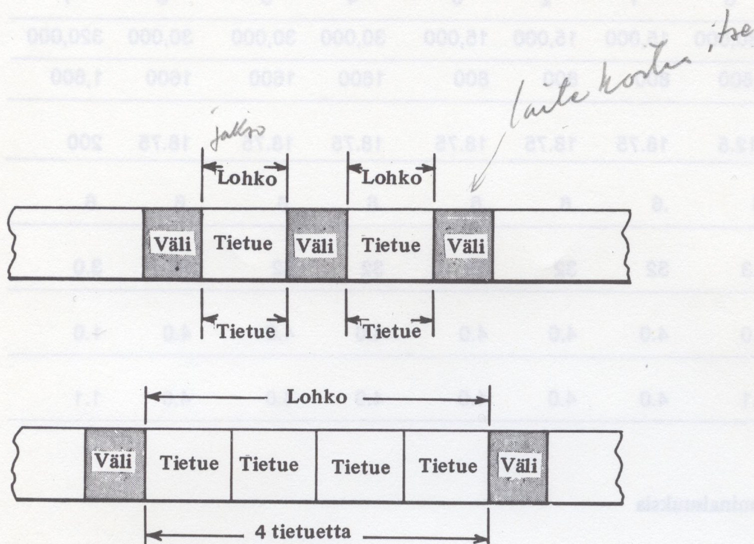
Nauhamerkin (erikoismerkki, jota edustaa heksadesimaaliluku 7F, kuva 39) avulla ilmoitetaan tiedoston loppuminen (kuva 85). Useimmat tietokoneet pystyvät lukemaan ja kirjoittamaan nauhamerkin.



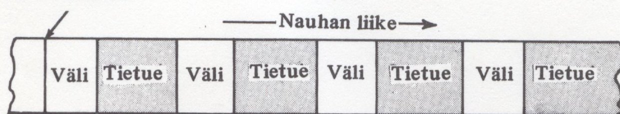
Kuva 84. Nauhakasetti

Magneettinauhayksiköiden ominaisuuksia

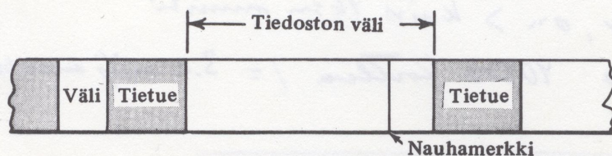
Tärkeimmät nauhayksiköiden suorituskykyyn vaikuttavat tekijät ovat nauhan liikkumisnopeus sekä tietojen tallennuksessa käytettävä merkkitiheys. Nämä kaksi tekijää määräävät sellaiset tärkeät ominaisuudet kuten tietojen siirtonopeuden ja hakuajan (gap time). Nauhayksiköt eroavat toisistaan lohkovälin pituuden, tarkistusmenetelmän, tarkistuksen laajuuden ja tallennetun tiedon suojaamisen suhteen. Kuvassa 86 on esitetty eräitä pääeroavuuksia eri nauhayksiköiden ominaisuuksista.



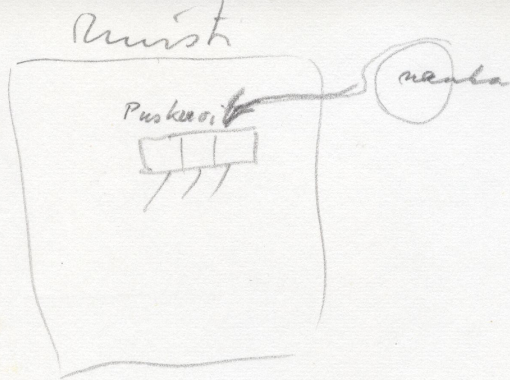
Magneettinauhalla tietoyksiköt eli lohkot erotetaan toisistaan lohkovälillä (lohkoa edeltävä ja sitä seuraava tyhjä osa nauhaa). Lohko saattaa sisältää yhden tai useampia loogisia tietueita.



Lohkoväli, jota seuraa tietty erikoismerkki (nauhamerkki) ilmoittaa tiedoston loppua. Nauhamerkin kirjoitus tapahtuu erityisellä käskyllä tiedoston viimeisen lohkon perään.



Kuva 85. Lohkon ja tiedoston lopun merkintä magneettinauhalla



	2401			2415						2420
	Malli 4	Malli 5	Malli 6	Malli 1	Malli 2	Malli 3	Malli 4	Malli 5	Malli 6	Malli 7
Tiedon siirtonopeus (tavua sekunnissa)	60,000	120,000	180,000	15,000	15,000	15,000	30,000	30,000	30,000	320,000
Tiheys (tavua tuumaa kohti)	1,600	1,600	1,600	800	800	800	1600	1600	1600	1,600
Nauhan nopeus (tuumaa sekunnissa)	37.5	75.0	112.5	18.75	18.75	18.75	18.75	18.75	18.75	200
Lohkoväli (tuumaa)	.6	.6	.6	.6	.6	.6	.6	.6	.6	.6
Lohkovälin ohitusaika (millisekunteja)	16.0	8.0	5.3	32	32	32	32	32	32	3.0
Takaisinkelausaika (minuutteja), uudelleen asetus mukaanluettuna	3.0	1.4	1.0	4.0	4.0	4.0	4.0	4.0	4.0	1.0
Takaisinkelaus ja nauhan poisto (minuutteja)	2.2	1.5	1.1	4.0	4.0	4.0	4.0	4.0	4.0	1.1

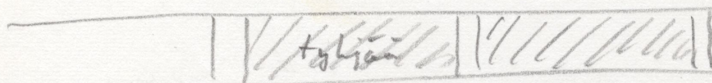
Kuva 86. IBM 2400-sarjan magneettinauhayksiköiden ominaisuuksia

Kelan nauha 2,400 jalkaa

reikäkortille 80

tallennustiheys/ 1 600
800
0,6"
2400'

yhtäjaksoinen tieto koko nauhalle 46 milj. tavua, on > kuin tk:n muisti
Jos ei tehdä jaksotusta mahtuu nauhalle 40500 korttia (= 3.2 milj merkkiä



1 kortti
käyttösuhde n 7%

jaksotus 10 korttia x 80 = 800 .. 247000 korttia
19.8 milj. merkkiä
43% käyttösuhde

-"- 25 -"- x 80 = 2000 375000 korttia
30 milj merkkiä
65% käyttösuhde

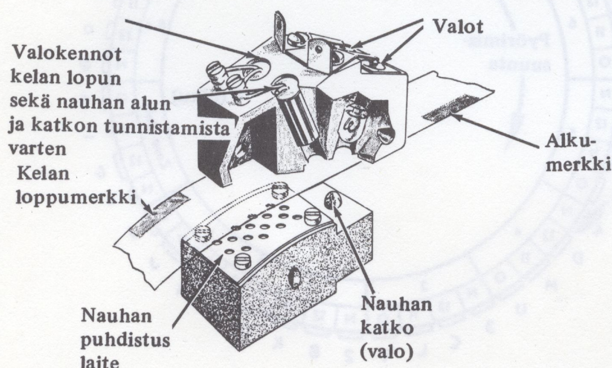
mitä pitemmät jaksot, sen isompi muisti on tarpeen

Tietojen siirron maksiminopeus ja todellinen nopeus

Koska nauhalla olevia tietolohkoja erottaa toisistaan lohkoväli, lohkon lukuun tarvittavaan kokonaisaikaan on otettava mukaan myös se aika, joka kuluu tämän tyhjän välin ohittamiseen. Tätä nimitetään hakuajaksi. Hakuaika kullakin nauhayksiköllä (kuva 86) määräytyy nauhan nopeuden ja lohkovälin pituuden perusteella. Hakuaika on tekijä, joka on otettava huomioon nauhayksikön todellista merkkinopeutta määritettäessä.

Magneettinauhan alku- ja loppumerkit

Magneettinauhayksikön syöttömekanismi edellyttää, että nauhan alussa ja lopussa on jonkin verran tyhjää tilaa. Nauhan varsinaisen tieto-osan alku ja loppu on merkitty selvästi havaittavin heijastavin liuskoin. Nauhayksikön valokennot pystyvät tunnistamaan nämä merkit joko alkumerkiksi (load point) josta nauhan luku tai kirjoitus voidaan aloittaa, tai loppumerkiksi (end-of-reel), johon nauhan kirjoitus päättyy. Lukutoiminnassa nauhan loppumerkkiä ei tunnisteta. Nauhamerkki (tape mark) ilmaisee tässä tapauksessa nauhan loppumisen (kuva 87).

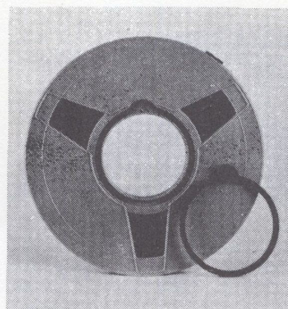


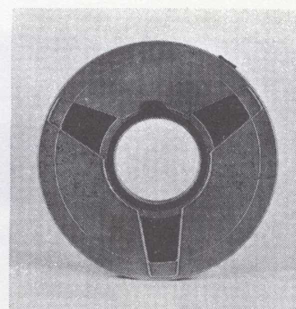
Nauhakelan alku ja loppu merkitään pienillä alumiinikalvon palasilla, jotka kiinnitetään nauhan päällystämättömälle puolelle. Uusissa nauhoissa merkit ovat valmiiksi paikoillaan. Magneettinauhayksikön valokennot suorittavat näiden merkkien tunnistuksen, samoin valokennojen avulla havaitaan rikkonainen nauha.

Kuva 87. Valon avulla tunnistettavat nauhamerkit

Tiedoston suojaaminen

Koska nauhalle kirjoittaminen automaattisesti tuhoaa sille aikaisemmin tallennetun tiedon, on välttämätöntä, että nauha voidaan suojata vahingossa tapahtuvalta tuhoamiselta. Tätä varten on käytettävissä nauhan 'suojauslaite' (kuva 88).





Normaalinauhoilla tiedoston suojaus voidaan hoitaa nauhakelassa olevaan uraan sopivan muovirenkaan avulla. Kun tämä rengas on paikoillaan, nauhalle voidaan kirjoittaa ja siltä lukea. Jos se on poistettu, nauhalta voidaan lukea täysin normaalisti, mutta sille ei voida kirjoittaa. Tällä tavoin nauha voidaan suojata päällekirjoittamiselta.

Kuva 88. Magneettinauhojen suojaus

POIMINTAMUISTILAITTEET

Tämän esityksen puitteissa poimintamuistilaitteet on luokiteltu muistien joukkoon, jossa ne myös on käsitelty (ks. jakso 'Muistilaitteet'). Aivan yhtä hyvin niitä voidaan kuitenkin pitää syöttö- ja tulostuslaitteina.

REIKÄNAUHAN LUKIJA

Reikänauhan lukija (kuva 89) pystyy lukemaan 5-, 6-, 7- ja 8-kanavaista reikänauhaa, jossa tieto esitetään lävistettyjen reikien avulla samaan tapaan kuin reikäkorteilla. Luvun maksiminopeus on 1000 merkkiä sekunnissa. Nauhan liikkuessa lukupään kautta tunnistetaan lävistetyt reiät (tai niiden puuttuminen) ja lähetetään ne sähköimpulsseiksi muunnettuina tietokoneeseen, jossa niitä voidaan käsitellä tietoina.

Luvun oikeellisuus voidaan määrittää merkkikohtaisella pariteettitarkistuksella (kahdeksankanavaista nauhaa käytettäessä). Lukunopeus, joka vaihtelee 150-1000 merkkiin sekunnissa, riippuu lukijasta ja luettavien tietueiden pituudesta.

Varsinaista tietokonetta koskevan syöttötoiminnan nopeuttamiseksi voidaan tiedot ensin erillisenä toimenpiteenä siirtää magneettinauhalle (kuva 90).

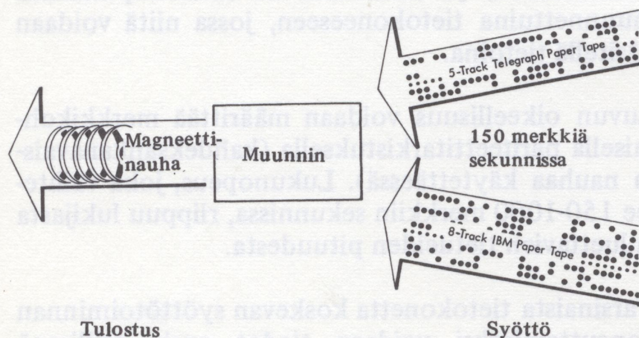


Kuva 89. IBM 2671 Reikänauhanlukija ja 2822 ohjain

Kun tiedot nyt voidaan lukea tietokoneeseen magneettinauhalta, on lukunopeus aivan toista luokkaa verrattuna suoraan reikänauhalta tapahtuvaan lukuun (ks. taulukko 86).

REIKÄNAUHAN LÄVISTIN

Tietokoneen tulostus voi tapahtua reikänauhalle reikänauhan lävistimen avulla. Keskusmuistista tulostettava tieto muunnetaan reikänauhakoodiksi



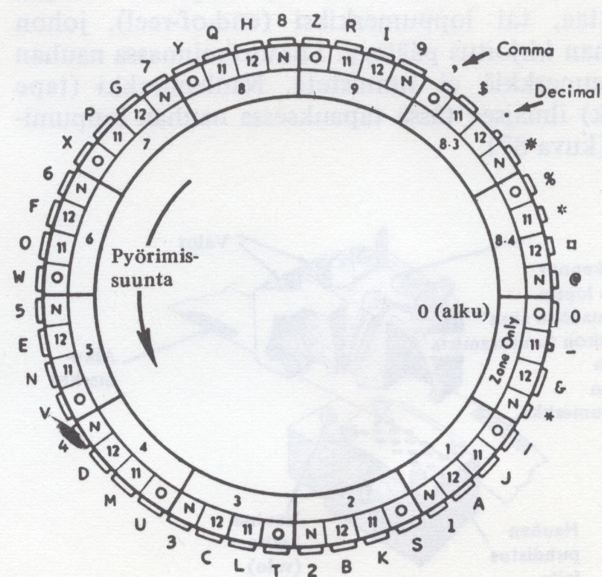
Kuva 90. Tietojen muunto reikänauhalta magneettinauhalle

ja lävistetään laitteen lävistysmekanismin läpi kulkevalle tyhjälle paperinauhalle. Tiedon oikeellisuus todetaan pariteettitarkistuksella (esim. kahdeksankanavainen koodi). Reikänauhan lävistystiheys on 10 merkkiä tuumalle ja lävistysnopeus 15-150 merkkiä sekunnissa.

RIVIKIRJOITTIMET

IBM-kirjoituslaitteiden avulla hoidetaan tietojen tulostus lomakkeille. Kirjoitusnopeus vaihtelee 10-2400 merkkiin sekunnissa.

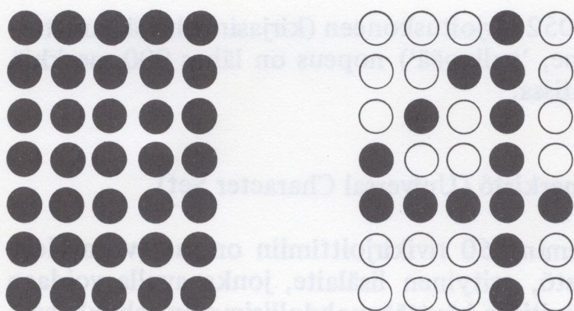
Tulostusyksikkönä toimiva rivikirjoitin ottaa keskusyksiköstä tulevat tiedot vastaan sähköimpulsseina. Impulssit johdetaan rivikirjoittimen virtapiireihin, joissa ne vuorostaan panevat liikkeelle tarvittavat kirjoituselementit. Kaikkiin kirjoituslaitteisiin kuuluu myös lomakkeensiirtolaitteisto, joka automaattisesti huolehtii paperin siirrosta kirjoituksen aikana.



Kuva 91. Kirjoituspyörä

Kirjoituslaitteet voidaan ryhmitellä kirjoitusmekanismin mukaan kirjoituspyörillä, lankakirjasinmatriisilla, kirjasinketjulla ja kirjasintangoilla varustettuihin rivikirjoittimiin. Omana ryhmänään voidaan pitää kirjoituskonetta.

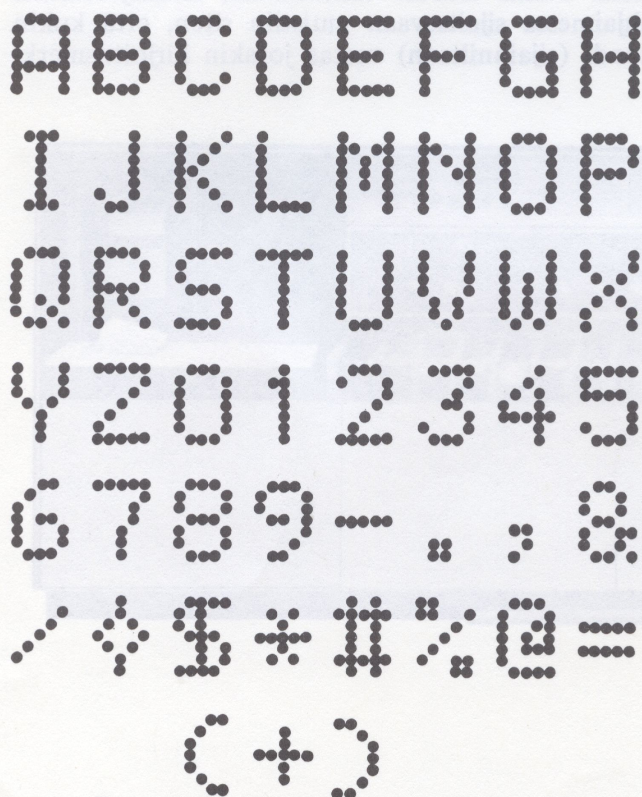
Pyöräkirjoittimessa on 120 kirjoituspyörää (kuva 91), joista jokainen sisältää 48 kirjoitusmerkkiä (kirjaimet, numerot ja erikoismerkit). Kirjoitettava



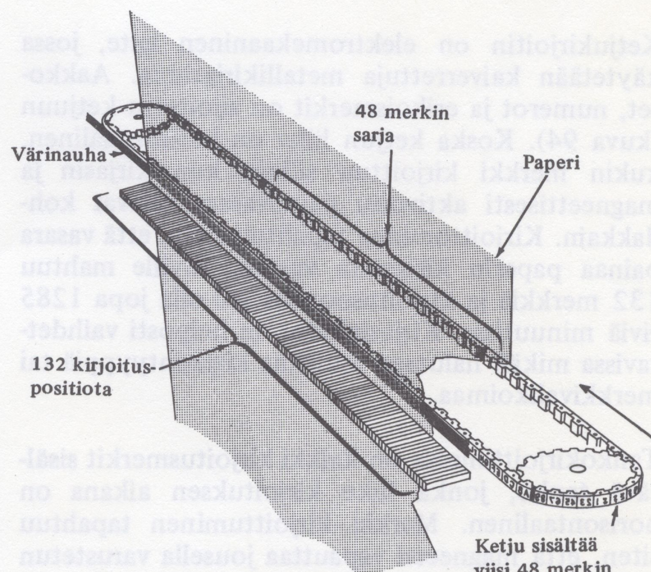
Kuva 92. 5 x 7 pistematriisi

tieto määrittää kunkin pyörän asennon etukäteen, koska koko 120 merkin pituinen rivi kirjoittuu yhdellä kertaa. Kirjoitusnopeus on 150 riviä minuutissa.

Lankamatriisikirjoittimessa merkit kirjoittuvat pienten lankojen päiden aikaansaamien pisteiden muodostamina kuvioina. Langat on järjestetty 5 x 7 langan matriiseihin (kuva 92). Lankakuvioiden avulla voidaan esittää 47 merkkiä (kirjaimet, numerot ja 11 erikoismerkkiä, kuva 93). Halutun merkin kirjoittamiseen tarvittavat langat painuvat kirjoitettaessa värinauhaa vasten, joka aiheuttaa merkin kirjoittumisen paperille. Rivin pituus on 120 merkkiä ja kirjoitusnopeus 500 tai 1000 riviä minuutissa kirjoittimen mallista riippuen.



Kuva 93. Lankakirjasimien kirjainkuviot



Kuva 94. Kirjasinketju



Kuva 95. IBM 1052 kirjoituskone

Ketjukirjoitin on elektromekaaninen laite, jossa käytetään kaiverrettuja metallikirjasimia. Aakkoset, numerot ja erikoismerkit on sijoitettu ketjuun (kuva 94). Koska ketjun liike on horisontaalinen, kukin merkki kirjoittuu silloin kun kirjasin ja magneettisesti aktivoitu painovasara tulevat kohdakkain. Kirjoittuminen tapahtuu siten, että vasara painaa paperin kirjasinta vasten. Riville mahtuu 132 merkkiä ja kirjoitusnopeus voi olla jopa 1285 riviä minuutissa. Kirjasinketju on helposti vaihdettavissa mikäli halutaan muuttaa kirjasintyyppiä tai merkkivalikoimaa.

Tankokirjoittimessa on kaikki kirjoitusmerkit sisältävä tanko, jonka liike kirjoituksen aikana on horisontaalinen. Merkin kirjoittuminen tapahtuu siten, että magneetti vapauttaa jousella varustetun vasaran, jolloin kirjasin painuu värinauhaa ja paperia vasten. Tällaisilla kirjoittimilla suurin nopeus on 240 merkkiä minuutissa.

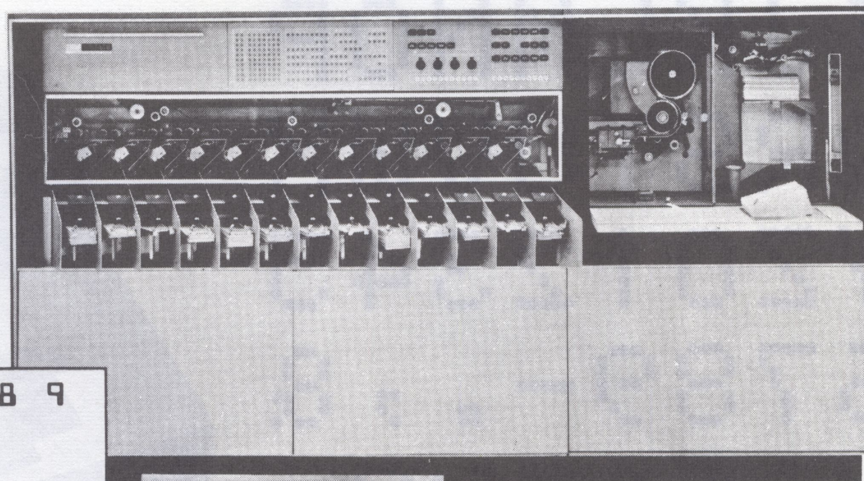
Tulostuslaitteena voidaan käyttää myös kirjoituskonetta (kuva 95). Tulostuslaitteena oleva kirjoituskone eroaa tavallisesta kirjoituskoneesta pääasiallisesti vain siinä, että kirjoitustoiminnan ohjauksesta ym. huolehtii ohjelma. Kirjoitusnopeus on noin 600 merkkiä minuutissa. Rivin siirto ja vaunun palautus ovat automaattisia toimintoja. IBM 1052 kirjoituskoneen (kirjasinkehällä varustettu kone, 'pallopää') nopeus on lähes 900 merkkiä minuutissa.

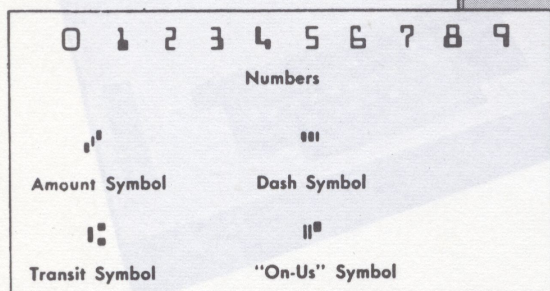
Yleismerkistö (Universal Character Set)

Systeemin/360 rivikirjoittimiin on saatavissa yleismerkistö, erityinen lisälaite, jonka avulla voidaan rivikirjoitinta käyttää mahdollisimman tehokkaasti. Tarvittaessa voidaan rivikirjoitin varustaa minkälaisella kirjasinvalikoimalla tahansa (maksimi on 240 erilaista merkkiä ketjussa). Ketjun merkkeihin voidaan määrittää mikä tahansa EBCDIC-bittiyhdistelmä (poisluettuina tyhjät merkit eli heksadesimaaliset 00 ja 40).

Käytettävä kirjasinketju voidaan muotoilla sellaiseksi kuin käyttäjä haluaa, eli merkkivalikoimaan voidaan sisällyttää vain ne merkit joita tarvitaan. (Kirjasinketjuja on nykyään saatavissa hyvinkin monen tyyppisiä).

Käyttäjä voi tämän jälkeen itse määrittää mikä bittiyhdistelmä vastaa mitäkin merkkiä kirjasinketjussa. Nämä koodit tallennetaan rivikirjoittimen ohjaimessa sijaitsevaan muistiin siten, että kukin koodi (sijainniltaan) vastaa jotakin kirjoitusmerk-





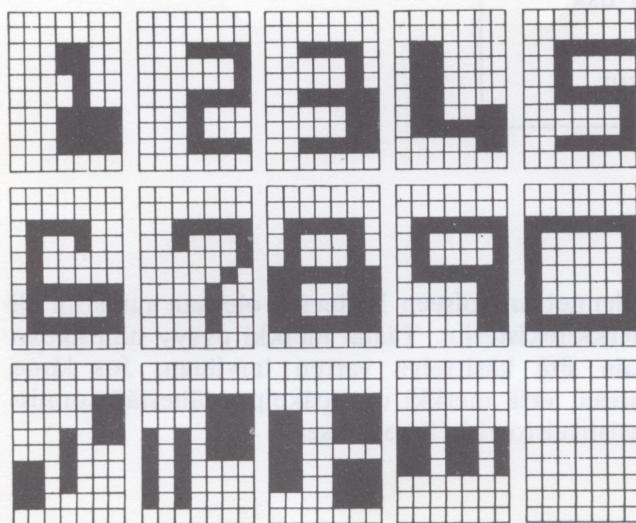
Kuva 96. IBM 1419 magneettimerkkien lukija ja magneettimusteella kirjoitettuja merkkejä

kiä (muisti sisältää 240 merkkipaikkaa kuten kirjasinketjukin). Riviä kirjoitettaessa kirjoittuu aina se kirjasinketjun merkki, joka vastaa ohjaimen muistissa olevaa koodia.

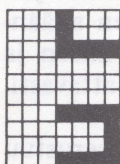
Rivikirjoittimissa, joissa ei tällaista lisälaitetta ole, rivin kirjoittamiseen kuluva aika on vakio. Lisälaitteella varustetuissa rivikirjoittimissa aika on vaihteleva (ja yleensä pienempi kuin edellisissä) ja rivin kirjoittamiseen kuluva aika katsotaan päättyneeksi kun ko. rivin viimeinen merkki on löytynyt.

MERKINLUKULAITTEET

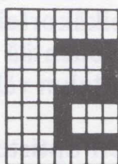
Erittäin nopeiden automaattisten tietojenkäsittelysysteemien tullessa markkinoille heräsi myös ajatus sellaisten syöttölaitteiden kehittämisestä, jotka pystyisivät lukemaan myös ihmisen ymmärtämää



Kukin lukupään ohittava merkki havaitaan ja tutkitaan. Lukijan tunnistusjärjestelmä tutkii täyttääkö merkki tietyt muodolle asetetut vaatimukset. Jos merkki ei lukupään ohittaessaan sijaitse täsmälleen siinä, missä sen pitäisi, tunnistusmatriisiin lähetettyjen signaalien perusteella muodostuu tietyllä tavalla käännetty merkki. Lukija muotoilee tällaisen käännetyn merkin automaattisesti siirtämällä sitä pystysuunnassa tunnistusta varten.



Käännetty merkkikuvio



Uudelleen muotoiltu merkki

Kuva 97. E13B-merkkikuviot

tekstiä. Vuosittain yli 13 miljardin shekin käsittelyä - puhumattakaan muista lomakkeista (vakuutusmaksut, sähkö- ym. laskut) - voitaisiin huomattavasti nopeuttaa tällaisten laitteiden avulla. Nykyään olemassa oleva laitteisto voidaan jakaa kahteen ryhmään - magneettimerkkien lukijat ja optiset lukijat.

MAGNEETTIMERKKIEN LUKIJAT

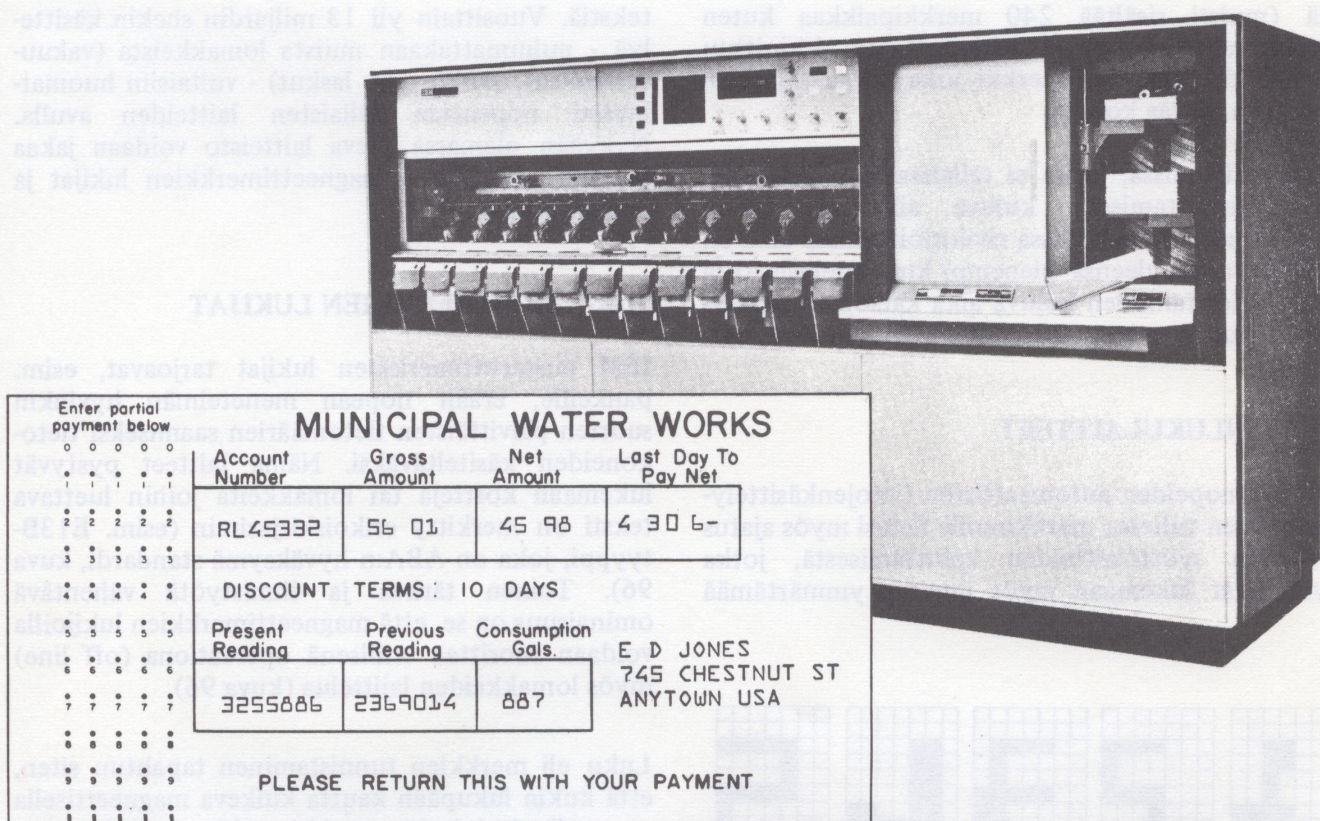
IBM magneettimerkkien lukijat tarjoavat, esim. pankeille, erään nopean menetelmän hyvinkin suurten päivittäisten tietomäärien saamiseksi tietokoneiden käsiteltäväksi. Nämä laitteet pystyvät lukemaan kortteja tai lomakkeita joihin luettava teksti on merkitty erikoiskirjasimin (esim. E13B-tyyppi, joka on ABA:n hyväksymä standardi, kuva 96). Toinen tärkeä ja ihmistyötä vähentävä ominaisuus on se, että magneettimerkkien lukijoilla voidaan suorittaa erillisenä operaationa (off line) myös lomakkeiden lajittelua (kuva 96).

Luku eli merkkien tunnistaminen tapahtuu siten, että kukin lukupään kautta kulkeva magneettisella musteella kirjoitettu merkki tutkitaan ja kymmenen tietokanavaa lähettää merkin aiheuttamat signaalit eräänlaiseen muistilaitteeseen, jota nimitetään merkkimatriisiksi. Matriisi on suuruudeltaan 70 'positiota' - yksi jokaista magneettisen merkin havaintopistettä kohti - ja kun lomakkeet kulkevat lukupään kautta, aiheuttaa hyväksyttävän signaalin puuttuminen kunkin merkin kohdalla tutkittavista havaintopisteistä sen, että vastaavaan positioon tallennetaan 0-bitti. Positiivinen signaali (joka siis merkitsee sitä, että lukupään kohdalla on magneettimusteella merkitty osa merkkiä) taas aiheuttaa 1-bitin tallentumisen vastaavaan positioon. Matriisiin tallentuva bittiyhdistelmä on myös nähtävissä koneen ohjaustaulun valoina.

Kun koko merkki on ohittanut lukupään ja sen kaikki osat on tutkittu, merkin muodostama kuvio on tallentunut matriisiin 0- ja 1-bittien yhdistelmänä (kuva 97). Oikeellisuuden toteamiseksi lukija tarkistaa automaattisesti jokaisen luetun merkin.

Kuvassa 97 esitettyjen 14 merkin osalta on tutkimus osoittanut, että kutakin merkkiä kohti on tuhansia hyväksyttäviä 0- ja 1-bittien yhdistelmiä vaikka lomakkeelle painettu merkki olisi puutteellinenkin.

Kun lukija on todennut merkin oikeaksi, se



Kuva 98. IBM 1428 aakkosnumeerinen optinen lukija



Kuva 99. IBM 2260 näyttölaite

tallennetaan toiseen laitteessa olevaan muistiin eli merkkirekisteriin. Täällä merkki pysyy niin kauan kuin sitä käsittelyä varten tarvitaan. Jos kone toteaa, että merkki on epäkelpo, lähettää tunnistuslaitteisto virheilmoituksen.

OPTISET LUKIJAT

Optiset lukijat pystyvät lukemaan lomakkeilta numeroita, kirjaimia (yleensä vain isoja kirjaimia) sekä tiettyjä erikoismerkkejä, ja siirtämään lukemansa tiedot tietokoneen käsiteltäviksi. Tällaisten laitteiden avulla voidaan eliminoida alkutietojen muunto reikäkorteille tai -nauhalle ennen sisäänlukua ja pienentää alkutositteiden syntymisen ja niiden käsittelyn välistä aikaa varsin huomattavasti.

Vanhempien optisten lukijoiden toiminta perustuu pyörivään rumpuun, joka kuljettaa lomakkeen syöttölokerosta optisen tutkainaseman ohi. Tutkain sisältää voimakkaan valolähteen ja linssijärjestelmän, jonka avulla lomakkeelta heijastuvat tummat ja vaaleat kuviot erotetaan toisistaan.

Nämä kuviot luetaan pienistä pisteistä muodostuvana joukkona ja muunnetaan sähköisiksi signaaleiksi. Jos optisesti luettu merkki vastaa jotakin lukijan tunnistusjärjestelmään kuuluvaa merkkiä, se tallennetaan ja siirretään edelleen tietokoneen keskusyksikköön käsittelyä varten. Merkin luku ja tunnistus ovat automaattisia toimintoja, joiden suoritusaika on sekunnin murto-osa.

Optiset lukijat voivat tunnistaa tavallisella lyijykynällä tehtyjä viivamerkintöjä. Menetelmä on hieman samankaltainen kuin vanhempi ns. marksensing. Viivan merkitys tai arvo määräytyy sen perusteella missä kohden se ko. sarakkeessa sijaitsee. Tyypillisenä sovellutuksena voidaan mainita sähkömittarien lukemamerkinnät ja varsinainen sähkölaskutus.

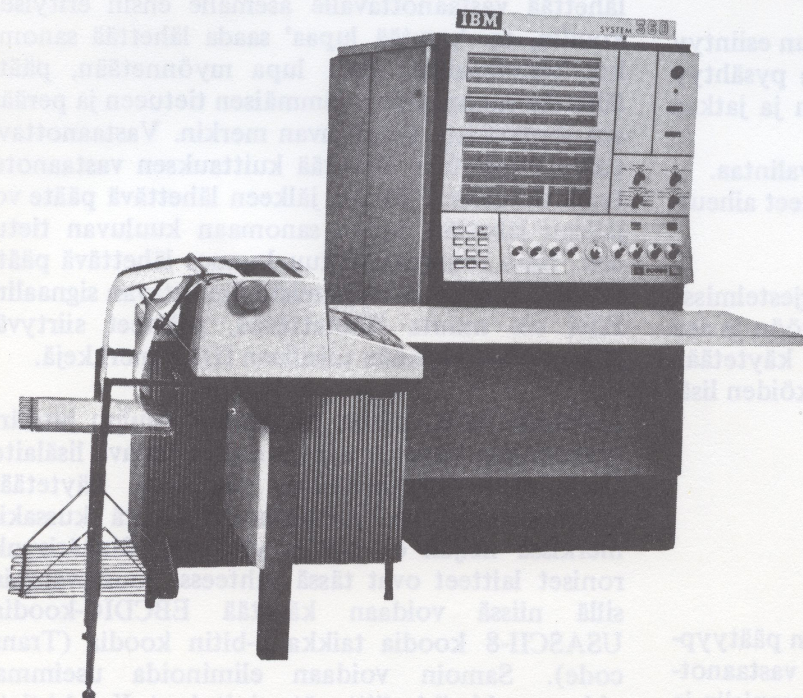
NÄYTTÖLAITTEET

Joihinkin tietojenkäsittelyjärjestelmiin kuuluu laitteita, joilla tietoja voidaan esittää visuaalisesti, kuvana, tulostamatta niitä paperille. Eräs tällaisista laitteista on Systeemiin/360 liitettävissä oleva IBM 2260 näyttölaite (kuva 99). Tietojen siirto keskusyksikön ja näyttölaitteen välillä tapahtuu 2560 merkin sekuntinopeudella. Paitsi 'normaalina' syöttö- ja tulostuslaitteena ja tietokoneen läheisyydessä sijaitsevana kyselyasemana, voidaan näyttölaitetta käyttää myös etäällä sijaitsevana terminaalina, jolloin se yhdistetään tietokoneeseen puhelinlinjojen välityksellä.

Tietojen siirtonopeus on tässä tapauksessa 120 tai 240 merkkiä sekunnissa linjoista riippuen. Kun tieto on siirretty näyttölaitteeseen, se säilyy kuvaputken pinnalla niin kauan kuin halutaan.

Näyttöalueen (suuruudeltaan 4 x 9 tuumaa) kapasiteetti on 6 tai 12 riviä, pituudeltaan 40 tai 80 merkkiä, ja kokonaismerkkimäärä vaihtelee siis 240:n ja 960:n välillä riippuen näyttölaitteiden ohjaimen mallista. Vakiomerkistö sisältää aakkoset ja numerot sekä 25 erikoismerkkiä. Merkkikuvio perustuu 5 x 7 pisteen matriisiin, jolloin tulostus ulkonäöltään on samanlaista kuin eräillä rivikirjoittimilla (kuva 92).

Tietojen syöttöä varten on IBM 2260:ssä näppäinpöytä. Näppäimistöllä kirjoitettu tieto siirtyy 2848-ohjaimen puskuriin, ja välittömästi myös näyttölaitteen kuvaputkelle visuaalista tarkistusta varten. Ennen kuin kuvaputkella oleva tieto lähetetään tietokoneeseen, se voidaan korjata (kokonaan tai osittain) tai pyyhkiä pois kokonaan.



Kuva 100. IBM Systeemin/360 mallin 30 ohjauspöytä

Tulostus tapahtuu normaalisti ohjelman valvonnassa. Tulostettava tieto voidaan kirjoittaa 'puhtaalle' näyttöalueelle tai lisätä siellä jo valmiiksi oleviin tietoihin. Paitsi kuvaputkelle voidaan tulostus ohjata IBM 1053 rivikirjoittimelle (joka voidaan liittää näyttölaitteen ohjaimeen ilman erikoislaitteita) samassa muodossa kuin kuvaputkellekin.

Toinen näyttölaite on tyyppimerkinnältään IBM 2250, jota on käsitelty jaksossa 'TIETOJEN ESITTÄMINEN', kappaleessa 'Visuaalinen tulostus'.

OHJAUSPÖYTÄ

Tietojenkäsittelyjärjestelmän ohjauspöydän (kuva 100) avulla operaattori voi ohjata järjestelmää ja valvoa sen toimintaa. Näppäimiä, kytkimiä, äänihälytysmerkkejä ja tarkkailupöydän valoja käyttäen operaattori voi:

1. Käynnistää ja pysäyttää tietokoneen.
2. Käsin ohjaamalla sijoittaa tietoa sisäiseen muistiin ja saada muistissa oleva tieto näkyviin valojen avulla.
3. Määrätä sisäisten sähköisten kytkimien tila.
4. Määrätä tiettyjen sisäisten rekistereiden sisällöt.
5. Muuttaa toimintatapaa siten, että kun esiintyy harvinainen tilanne, tietokone joko pysähtyy tai ilmoittaa tilanteen olemassaolon ja jatkaa toimintaansa.
6. Muuttaa syöttö/tulostusyksiköiden valintaa.
7. Nollata tietokoneen, kun virhetilanteet aiheuttavat sen pysähtymisen.

Joissakin suurissa tietojenkäsittelyjärjestelmissä pääohjauspöytä on liitetty keskusyksikköön ja sen lisäksi on erillisiä ohjauspöytiä, joita käytetään huoltotoimintaan ja syöttö/tulostusyksiköiden lisäohjaukseen.

PÄÄTTEET

Kaukosiirtolaitteet voidaan jakaa kahteen päätyyppiin sen mukaan miten ne lähettävät ja vastaanottavat tietoja, nimittäin ns. 'start/stop'-tyyppisiin ja synkronisiin (STR) tietojen siirtolaitteisiin. IBM 1050, 1030, 1060 ja 1070 järjestelmät kuuluvat start/stop-tyyppisiin. Näiden toiminta perustuu asynkroniseen tietojen lähettämiseen (jossa jokainen merkki tahdistetaan erikseen erityisen aloitusmerkin avulla). Jokaista käytettävän koodin elementtiryhmää (joka vastaa merkkiä) edeltää aloitussignaali, jonka tehtävänä on valmistella vastaanottomekanismi ottamaan vastaan ja rekisteröimään sitä seuraava merkki. Lisäksi jokaista merkkiä seuraa erityinen loppusignaali, joka puolestaan vapauttaa vastaanottolaitteiston seuraavaa merkkiä varten.

Synkronisiin laitteisiin kuuluvat sellaiset laitteet kuin IBM 1009, 1013, 7701, 7702, 7710, 7711. Synkroninen tietojen lähetys merkitsee sitä että lähetettävän sanoman merkit seuraavat toisiaan välittömästi ilman mitään alku- tai loppumerkkejä.

Tavallisin yhteysmuoto synkronisten päätteiden välillä on vuorosuuntainen (half-duplex). Päätteet ovat jatkuvasti yhteydessä toisiinsa. Ellei päätteellä ole mitään lähetettävää, se on 'joutokäynnillä' ja lähettää toiselle päätteelle tätä tilaa kuvaavaa signaalia (erikoismerkkiä) n. 1,5 sekunnin ajan.

Sillä välin toinen pääte tahdistuu vastaanottamaan ko. signaalia. Kun mainittu aika, n. 1,5 sekuntia on kulunut, päätteet vaihtavat osia (siis ensimmäinen pääte siirtyy 'kuuntelulle' ja toinen lähetykseen). Kun päätteellä on lähetettävänä jokin sanoma, se lähettää vastaanottavalle asemalle ensin erityisen koodin, ts. 'pyytää lupaa' saada lähettää sanoma ko. terminaalille. Kun lupa myönnetään, pääte lähettää sanomansa ensimmäisen tietueen ja perään sen loppumista ilmoittavan merkin. Vastaanottava pääte puolestaan lähettää kuittauksen vastaanotetusta tietueesta. Tämän jälkeen lähettävä pääte voi jälleen lähettää uuden sanomaan kuuluvan tietueen. Näin toiminta jatkuu kunnes lähettävä pääte lähettää sanoman loppumista ilmoittavan signaalin. Ellei ole muuta lähetettävää, päätteet siirtyvät jälleen lähettelemään toisilleen IDLE-merkkejä.

Joihinkin synkronisiin päätteisiin kuuluu binäärisynkronisen tietojen siirron mahdollistava lisälaite. Useimmissa synkronisissa päätteissä käytetään erikoiskoodia (so. kahdeksasta bitistä kussakin merkissä neljän on oltava ykkösiä). Binäärisynkroniset laitteet ovat tässä suhteessa joustavampia, sillä niissä voidaan käyttää EBCDIC-koodia, USASCII-8 koodia taikka 6-bitin koodia (Transcode). Samoin voidaan eliminoida useimmat ohjausmerkkeihin liittyvät rajoitukset. Kaikki tieto lähetetään binäärisenä tietojonona, josta se sitten kooditetaan kulloinkin käytettäväksi merkistöksi. Binäärisynkroniseen tietojen siirtoon tarvittava

erikoislaite on kehitetty IBM 2780 laitteistoa varten.

Synkronisten päätteiden käyttämä tietojen siirtomenetelmä on paljon nopeampi ja tehokkaampi kuin start/stop-tyyppisissä päätteissä käytettävä. Useimmissa tapauksissa synkroninen tietojen siirto on yli 20 kertaa nopeampaa kuin start/stop-siirto.

TIETOJEN PUSKUROINTI

Kaikki tietojenkäsittelyprosessit sisältävät tietojen syötön, käsittelyn ja tulostuksen. Jokainen prosessin vaihe vie tietyn ajan. Tietokoneen käyttökelpoisuus on usein suoraan verrannollinen nopeuteen, jolla se voi suorittaa annetun tehtävän. Kaikki toiminnat, jotka eivät käytä hyödykseen keskusyksikön koko kapasiteettia, estävät koko järjestelmää toimimasta suurimmalla mahdollisella tehollaan. Ihanteellisissa tapauksissa eri syöttö- ja tulostuslaitteiden yhdistelmän ja laitteiden nopeuksien pitäisi olla siten sovitettuja, että keskusyksikkö olisi aina hyödyllisessä työssä.

Jokaisen tietojenkäsittelyjärjestelmän tehoa voidaan lisätä siihen määrään saakka, että syöttö-, tulostus- ja sisäiset käsittelytoiminnat voidaan osittain limittää toisiinsa tai suorittaa samanaikaisina toimintoina.

Syöttötiedot jaetaan erityisiin yksiköihin eli tietojen loogisiin yhdistelmiin, jotka sijoitetaan muistiin ohjelman valvonnan alaisina. Jokainen yksikkö käsitellään tavallisesti ennen seuraavan tietoyksikön lukemista. Joukko tulostustietoja saatetaan kehittää yhdestä syöttöyksiköstä tai useita syöttötietoja saatetaan yhdistää muodostamaan yksi tulos.

Kuvassa 101A esitetään perusajan vertailua syötön, käsittelyn ja tulostuksen välillä ilman toimintojen limitystä. Tämän tyyppisessä toiminnassa käsittely keskeytyy syöttö- ja tulostustoimintojen ajaksi. Tehokkuus on heikko, koska suurin osa keskusyksikön käytettävissä olevasta ajasta kuluu hukkaan.

Kuvassa 101B esitetään eräs mahdollinen aikavertailu syötön, tulostuksen ja käsittelyn välillä puskurointimenetelmää käytettäessä. Tiedot kerätään ensin erääseen ulkoiseen laitteeseen, jota sanotaan puskuriksi. Puskurin sisältö siirtyy keskusmuistiin ohjelman ohjaamana. Siirto vaatii vain murto-osan siitä ajasta, mikä tarvittaisiin tietojen lukemiseen suoraan syöttölaitteesta. Sillä aikaa kun tiedot kerääntyvät puskuriin, tietokoneessa voi tapahtua tiedon sisäistä käsittelyä. Samoin voidaan loppuun käsitellyt tiedot siirtää hyvin nopeasti keskusmuistista puskuriin. Tulostuslaite saa sitten käskyn tulostaa puskurin sisällön. Tulostuksen aikana keskusyksikkö on vapaa ja voi jatkaa muita töitä. Jos systeemiin on liitetty useita puskuroituja laitteita, luku, kirjoitus ja käsittely voivat tapahtua samanaikaisesti (kuva 101C).

Puskurointimenetelmien kehittäminen on johtanut siihen, että keskusmuistia käytetään primäärisenä puskurina. Tiedot otetaan syöttö/tulostuslaitteista tai lähetetään niihin sanoina tai kiinteän pituisina merkkiryhminä. Sanojen siirto tapahtuu automaattisesti käsittelyn aikana, mutta yhden sanan siirtoon kuluva aika on suhteellisen merkityksetön. Tämän ansiosta saadaan tietojen sisäinen käsittely osittain samanaikaiseksi sekä syöttö- että tulostustoimintojen kanssa. Pääetuna tästä kuitenkin on se, että käsiteltävän tiedon pituutta rajoittaa vain keskusmuistin käytännöllinen koko. Ulkoisia puskureita käytettäessä käsiteltävien tietojen pituutta rajoittaa puskurin kapasiteetti.

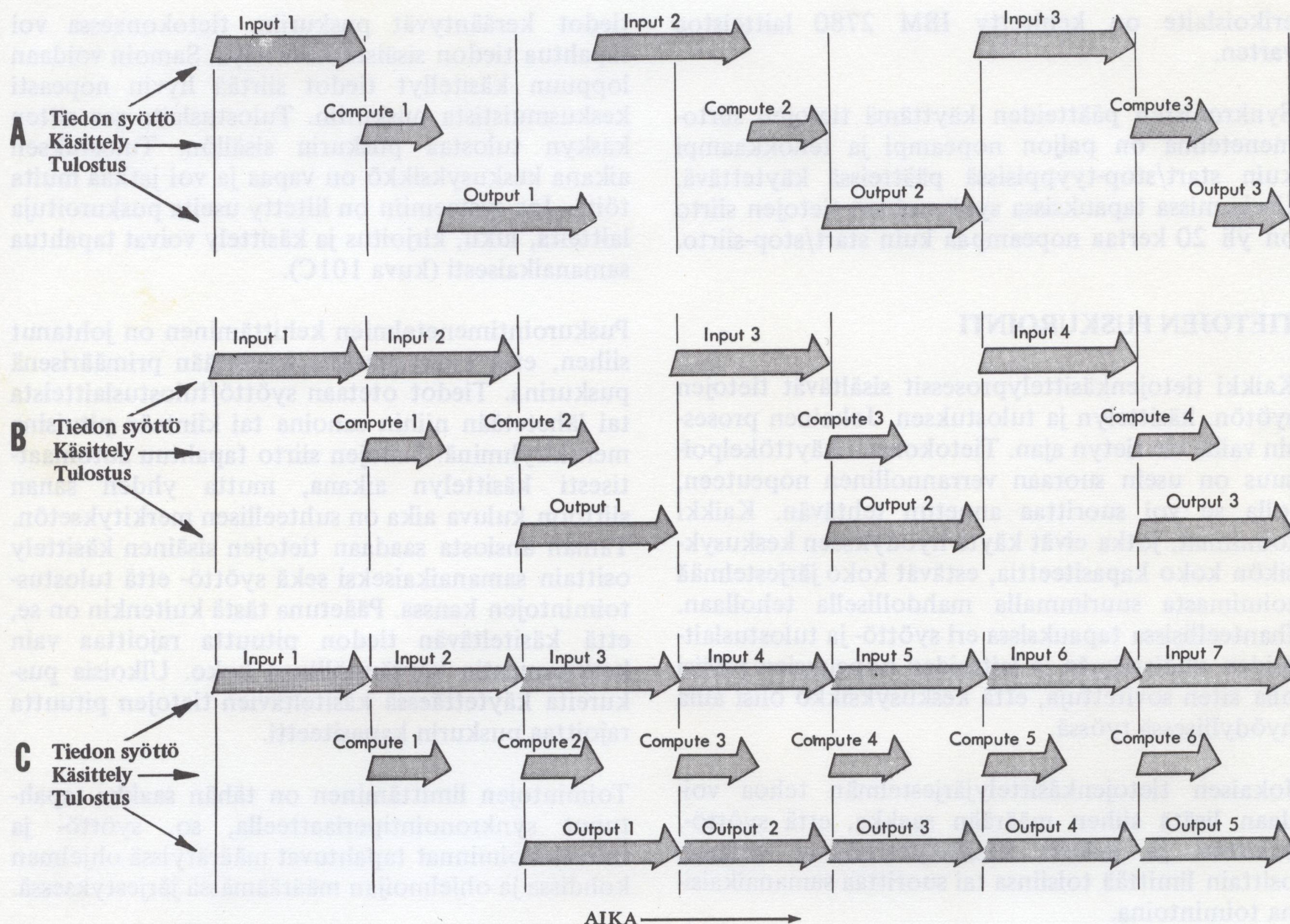
Toimintojen limittäminen on tähän saakka tapahtunut synkronointiperiaatteella, so. syöttö- ja tulostustoiminnat tapahtuvat määrätyissä ohjelman kohdissa ja ohjelmoijan määräämässä järjestyksessä.

Jotkut tietokoneet on rakennettu siten, että syöttö- ja tulostuslaitteet voivat automaattisesti keskeyttää tietojen käsittelyn, synkronointi ei ole tarpeellinen. Syöttö/tulostuslaite itse lähettää signaalin keskusyksikköön, kun se on valmis lukemaan tai tulostamaan. Keskusyksikkö vastaa näihin signaaleihin ja joko vastaanottaa syöttötiedot tai lähettää pyydetyt tiedot tulostettavaksi. Reaaliaikajärjestelmissä tämän tyyppinen toiminta on satunnaista ja ennalta-arvaamatonta.

Ongelmana on miten keskusyksikön aika käytetään. Vastaus on se, että sijoitetaan eri tehtävät jonoon ja annetaan niiden käyttöön keskusyksikön ylimääräinen aika. Juuri tässä on moniajon perusta - aihetta käsitellään yksityiskohtaisemmin jaksossa 'OHJELMOINTIJÄRJESTELMÄT'.

APUTOIMINNAT

Tietojenkäsittelyjärjestelmän syöttö/tulostustoiminnat ja tietojen siirtäminen tallennusmateriaalilta toiselle ovat suhteellisen hitaita keskusyksikön nopeuteen verrattuna. Apu- eli OFF-LINE-toimin-



Kuva 101. Tietojen puskurointi

nat tarjoavat menetelmän, jolla järjestelmään suoraan kytkemättömät laitteet voivat suorittaa muita toimintoja. Saavutettu etu on tietokoneen vapautuminen rutiiniluontoisista aikaa vievistä toiminnoista ja lisäajan saaminen keskusyksikössä tapahtuville laskentatoiminnoille ja tietojen käsittelylle.

Pääaputoiminnat ovat tietojen siirtäminen korteilta magneettinauhalle, magneettinauhalta korteille ja magneettinauhalta kirjoittaminen rivikirjoittimella.

Esimerkiksi kaikki tulostustiedot voidaan tallentaa magneettinauhoille, mikä on varsin nopea tietojen tallennustapa. Nauhalta tiedot voidaan sitten siirtää aputoiminnoilla korteille tai kirjoittaa rivikirjoittimella tekstinä paperille tietokoneen jatkaessa uusien tietojen käsittelyä.

Aputoimintojen merkitys on kasvanut siinä määrin, että nyt on mahdollista joissakin suuremmissa tietokoneissa käyttää pieniä tietojenkäsittelyjärjestelmiä niiden suorittamiseen.

OHJELMAN PERUSKÄSITTEISTÖ

Kun tietomateriaali on siirretty jollekin syöttövälineelle, voi tietokone ryhtyä huolehtimaan tietojen käsittelystä ja käsittelyn tuloksista. Tietokoneen sisällä suoritettavat toimenpiteet ovat kuitenkin ensin täsmällisesti ja yksityiskohtaisesti järjestelmän ominaisuuksien mukaisina toiminta-alkioina. Näitä toiminta-alkioita nimitetään käskyiksi.

Tietyn toimintasarjan, proseduurin toteuttamiseen tarvittavaa käskyjoukkoa nimitetään ohjelmaksi. Nykyisissä tietojenkäsittelyjärjestelmissä ohjelma tallennetaan ennen toteuttamista tietokoneen muistiin (siitä aikaisempi nimitys 'tallennettu ohjelma') jolloin myös käskyjen saanti tapahtuu elektronisella nopeudella.

KÄSKYT

Tietokone suorittaa jokaisen toimintansa käskyn ohjaamana. Käsky on keskusmuistissa oleva määrätty tietoyksikkö, jonka keskusyksikkö tulkitsee, kun toiminta on suoritettava.

Jos tiedot ovat koneen sisällä, käsky ohjaa konetta pääsemään niihin käsiksi. Jos on käytettävä jotakin laitetta, esimerkiksi magneettinauhayksikköä, käsky määrittää laitteen ja tarvittavat toiminnat.

Käskyt voivat muuttaa indikaattorin tilan, ne voivat siirtää tiedon muistipaikasta toiseen, ne voivat panna nauhayksikön kelaamaan nauhaa takaisin ja ne voivat muuttaa laskijan sisältöä. Muutamat käskyt voivat mielivaltaisesti tai johonkin koneen tai tiedon antamaan osoitukseen nojautuen määrätä seuraavan suoritettavan käskyn muistipaikan. Tällä tavoin on mahdollista muuttaa sitä järjestystä, jota kone tavallisesti noudattaa jokaista käskyä tai käskysarjaa suorittaessaan.

Käsky	Operandi
Valitse (Select)	Nauhayksikkö 200
Lue (Read)	Tietue muistipositioihin 1000-1050
Nollaa ja lisää (Clear and Add)	Positiossa 1004 oleva luku (nollattuun laskuriin)
Vähennä (Subtract)	Positiossa 1005 oleva luku laskurin sisällöstä
Tallenna (Store)	Tulos positioon 1051
Hyppää (Branch)	Positiossa 5004 olevaan käskyyn

Kuva 102. Käskyjä

Jokainen käsky (kuva 102) sisältää ainakin kaksi osaa:

1. Operaatio-osa, joka ilmoittaa suoritettavan toiminnan: lue, kirjoita, laske yhteen, vähennä, vertaa, siirrä tietoja jne.
2. Operandin, joka ilmoittaa tiedon tai määrättyyn toimintaan tarvittavan laitteen osoitteen.

Käskykierroksen aikana keskusyksikkö ottaa käskyn muistista ja analysoi sen. Operaatio-osa ilmoittaa suoritettavan toiminnan. Tämä tieto on koodattu siten, että sillä on jokin tietty merkitys koneelle. Esim. Systeemissä/360 kirjain 'A' merkitsee yhteenlaskua (add), 'C' vertailua (compare), SIO syöttö/tulostustoiminnan aloituskäskyä (Start Input/Output) ja TR muuntoa (translate). Muissa järjestelmissä käskykoodien merkkeinä käytetään muita kirjaimia ja numeroita.

Operandi sisältää toiminnan suorittamiseen tarvittavia lisämäärityksiä. Esimerkiksi aritmeettisissa toiminnoissa ilmoitetaan toisen muistissa olevan tekijän osoite. Syöttö- ja tulostuslaitteiden kyseessä ollen määrätään käytettävä yksikkö. Luettaessa ja kirjoitettaessa määrätään muistialue syötettäviä tai tulostettavia tietojaksoja varten, ellei sitä ole määrätty jo koneen suunnitteluvaiheessa.

Koska käskyt muistivälineiden ja -laitteiden suhteen ovat samassa asemassa kuin muutkin tiedot, on niiden koodijärjestelmänkin oltava sama. Joissakin tietokoneissa, esim. IBM 7090:ssä, käskyjen pituus on kiinteä (yksi sana). Toisissa taas käskyjen pituus on vaihteleva (esim. IBM 1401, 1410). Systeemissä/360 käskyjen pituudet ovat puolisana (2 tavua), sana (4 tavua) tai puolitoista sanaa (6 tavua) riippuen käskyn tyypistä ja sen vaatimista operandeista.

Muistissa ei yleensä ole mitään erityisesti käskyjä varten varattua aluetta. Useimmiten ohjelman käskyt muodostavat yhden kokonaisuuden ja ne sijoitetaan muistiin peräkkäisiin muistipaikkoihin siinä järjestyksessä, missä niiden toteutuskin tapahtuu. Käskyjen suoritusjärjestys voi kuitenkin luonnollisesti vaihdella haarautumiskäskyjen, tietojen laadun, systeemin ja laitteiden tilan, ulkopuolisten keskeytysten (esim. tietojen kaukokäsittelyoh-

jelmissa) tai muun sellaisen syyn takia. Ohjelman toteutuksen voi keskeyttää myös toinen ohjelma, joka prioriteetiltaan on aikaisempaa korkeampi jne.

Tietokoneen toimintajärjestys ohjelman kannalta on seuraava:
Keskusyksikkö hakee ensimmäisen käskyn joko ohjelmalle määritetyn alueen alusta tai alueelta, jonka osoite on annettu esim. koneen ohjaustaululta ja toteuttaa sen. Tämän jälkeen tapahtuu seuraava käskyn haku (edellistä käskyä seuraavasta muistipaikasta tai haarautumiskäskyn osoitteen määrittämästä muistipaikasta). Toiminta jatkuu samaan tapaan kunnes ohjelma on suoritettu loppuun tai ohjelmassa on kohdattu pysähtymiskäsky.

Kaksiosoitteiset käskyt

Joidenkin tietokoneiden, esim. Systeemin/360 käskyissä on kaksi osoiteosaa. Käskyn määrittämästä toiminnosta riippuen nämä osoitteet voivat esim. määrittää käytettävän laitteen sekä tiedon, johon toiminta kohdistuu, taikka kaksi tietokenttää. Ensimmäinen osoite voi määrittää käytettävän tulostuslaitteen ja toinen sen alueen, josta tietojen tulostus tapahtuu. Laskutoimituksissa osoitteet taas voivat määrittää tekijät, kertojan ja kerrottavan, jakajan ja jaettavan jne.

Tietyn toimintasarjan toteuttamiseen tarvitaan vähemmän kaksiosoitteisia käskyjä kuin yksiosoitteisia. Näin ohjelmointi helpottuu ja myös ohjelman tilantarve tietokoneen muistissa pienenee.

Käskyt ja tieto

Ainoa ero keskusmuistissa olevien käskyjen ja tietojen välillä on siinä ajassa, jolloin ne tuodaan keskusyksikköön. Jos tieto tuodaan keskusyksikköön käskyjakson aikana, se tulkitaan käskyksi. Jos se tuodaan keskusyksikköön jonkin muun jakson aikana, sitä pidetään tietona.

Tästä johtuen tietokone voi operoida omilla käskyillään, mikäli nuo käskyt otetaan muistista tietoina. Kone voidaan myös ohjelmoida muuttamaan omia käskyjään tehtävän käsittelyn aikana ilmaantuneiden tilanteiden mukaan. Juuri tämä käskyjen muuttamiskyky suo muistiin tallennettuja ohjelmia käyttäville järjestelmille melkein rajattoman joustavuuden ja niin sanotun loogisen kyvyn.

OHJELMAN LAATIMINEN

Ohjelman laatimiseksi on ohjelmoijan tunnettava ensinnäkin tehtävän puitteet muodostavan järjestelmän eri toiminnot. Toiseksi on tunnettava itse proseduuri, joka tullaan vaihe vaiheelta muuntamaan tietokoneen käskyiksi. Kolmanneksi, on luonnollisesti tiedettävä mitä tuloksia ohjelmalta edellytetään.

Ensimmäinen työvaihe on ohjelmoitavan sovellutuksen täydellinen tutkimus, analyysi. Tutkimuksen piiriin sisältyvät sekä olemassa olevat että suunnitellut proseduurit. Tähän vaiheeseen kuuluu yleensä työnkulku- ja ohjelmakaavioiden laatiminen, koska useimpiin tietojenkäsittelysovellutuksiin liittyy suuri määrä vaihtoehtoja, poikkeuksia ym. tilanteita, jotka vaativat oman käsittelynsä.

Kaikkia näitä mahdollisuuksia on hankala esittää sanallisesti. Sen sijaan erilaiset kuvat ja kaaviot tarjoavat suunnittelijalle ja ohjelmoijalle käyttökelpoisemman esitystavan. Seuraava kaavioita käsittelevä aihe koskettelee lähinnä järjestelmä- ja ohjelmakaavioita, eli yleisesti ottaen lohkokaavioita.

Lohkokaavion suurin etu on siinä, että se antaa aiheesta kokonaiskuvan yhdellä silmäyksellä. Se on graafinen esitys, joka antaa selvän kuvan järjestelmän proseduureista ja järjestelmän tietojen kulusta sekä näiden välisistä suhteista. Vastaavan 'kuvan' saaminen sanallisesta esityksestä on erittäin hankalaa - ohjelmoinnin perustaksi pelkkä sanallinen esitys ei ole riittävä. Lohkokaaviossa sen sijaan asetetaan sana ja symboli palvelemaan toinen toisiaan. Tekstillä voidaan selventää ja tarkentaa symbolin merkitystä ja päinvastoin. On kuitenkin luonnollista, että tällaisen kaavioesityksen käytön kommunikaatiovälineenä tekee mielekkääksi vasta yleisesti hyväksytty, standardoitu kuvakieli (kuva 103).

Piirtämisen apuvälineenä käytettävä mallilevy sisältää tarvittavat symbolit sekä järjestelmä- että ohjelmakaavioita varten. Kaavioiden ero selviää oikeastaan jo käytetystä nimityksestä. Järjestelmäkaavio esittää tietojen kulkua koko tietojenkäsittelysysteemissä - ohjelmakaavio taas kuvaa järjestelmän yhtä osaa, ohjelmaa.



FLOWCHARTING TEMPLATE

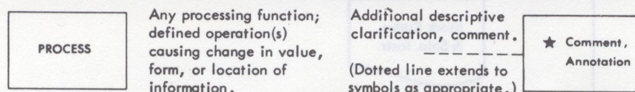
FORM X20-8020-1 U/M 01

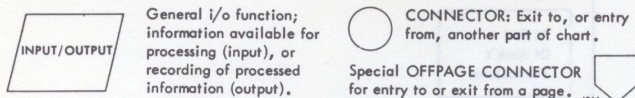
Symbols on this envelope—reflecting additions and changes—conform to the International Organization for Standardization (ISO) Draft Recommendation on Flowchart Symbols for Information Processing, and are consistent with the fewer symbols adopted by the U.S.A. Standards Institute (USASI). ISO usages beyond USASI specifications are identified (ISO). IBM usages beyond ISO specifications are three symbols—offpage connector, transmittal tape, keying—identified IBM.

★ Composite Symbols (preceded by a star) are those drawn by adding to or combining shapes provided by cutouts in the template.

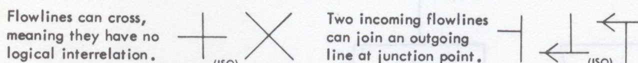
On this envelope, symbols are in three groups: (1) basic symbols; (2) processing and sequencing symbols related to programming; (3) input/output, communication link, and processing symbols related to systems.

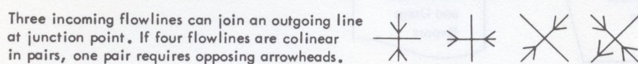
BASIC Symbols



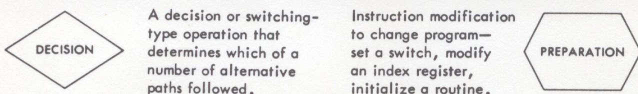


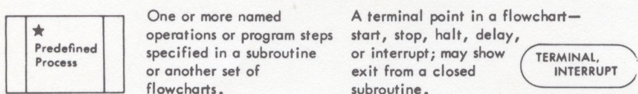
ARROWHEADS and Flowlines: In linking symbols, these show operations sequence and dataflow direction. Arrowheads required if path on any linkage is not left-to-right or top-to-bottom.



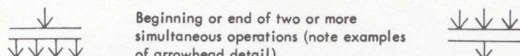


Symbols related to PROGRAMMING





Parallel Mode (ISO):

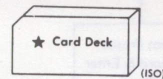


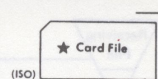
BASIC Symbols (shown at top) also are used in program flowcharting and in systems flowcharting (see other side of envelope).

Symbols related to SYSTEMS

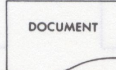
Input/output function in card medium (all varieties):

PUNCHED CARD





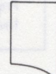
Other specific media:





TRANSMITTAL TAPE

Proof- or adding machine tape, or other batch-control info.



PUNCHED TAPE

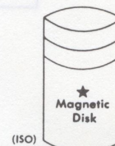
ONLINE STORAGE

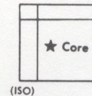
Input/output using any kind of online storage—magnetic tape, drum, disk.

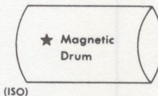
An operation using a key-driven device—such as punching, verifying, typing.

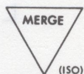
KEYING IBM

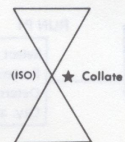
Other specific media for input/output functions:

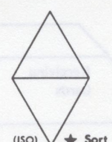


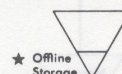












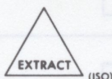
Combining two or more sets of items into one set.

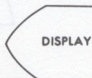
Removal of one or more specific sets of items from a set.

Merging with extracting; forming two or more sets of items from two or more other sets.

Arranging a set of items into sequence.

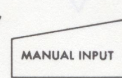
Storing offline, regardless of recorded medium.

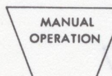




Information display by online indicators, video devices, console printers, plotters, etc.

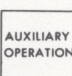
Information input by online keyboards, switch settings, pushbuttons.



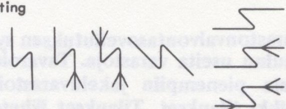


Any offline process (at "human speed") without mechanical aid.

Offline performance on equipment not under direct control of central processing unit.



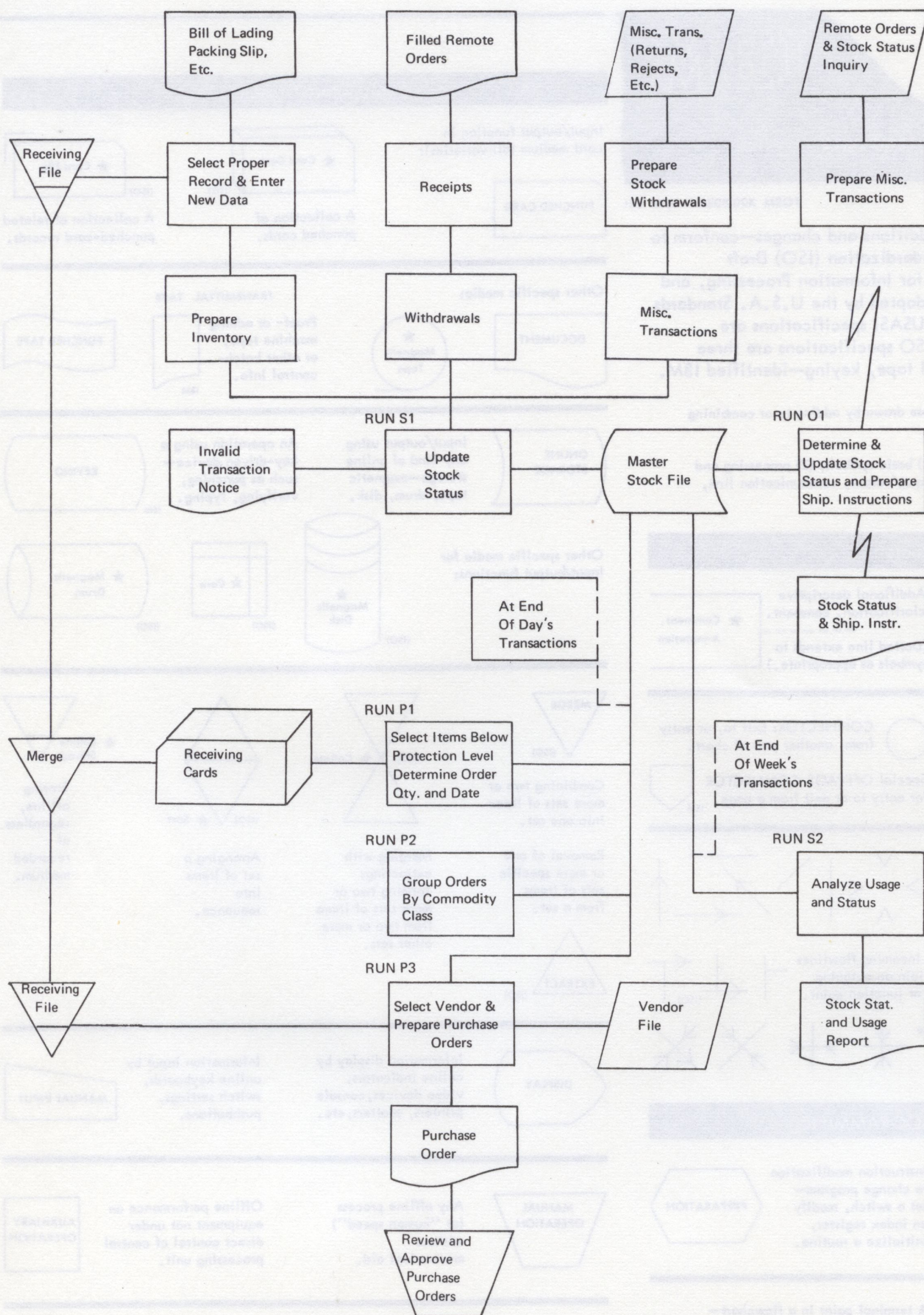
COMMUNICATION LINK: Function of transmitting information by a telecommunication link. (Vertical, horizontal, or diagonal, with arrowheads for clarity; bidirectional flow shown by two opposing arrowheads.)



BASIC Symbols (shown on other side of envelope) also are used in systems flowcharting.

Your IBM representative can give information about computerized flowcharting with the IBM System/360 Flowchart Program

Kuva 103. Ohjelma- ja systeemikaavioissa käytettävät symbolit



Varastonvalvontasovellutuksen systeemikaavio. Sovellutuksen piiriin kuuluu useita varastoja. Tavaroiden toimitus tapahtuu keskusvarastosta pienempiin jakeluvarastoihin, jotka myös ottavat vastaan kaikki tilaukset. Tilaukset lähetetään tietoliikenneverkoston avulla tietokonekeskukseen. Systeemin perustoiminnot ovat seuraavat:

1) **Varastotilanteen päivitys (ajo S1) tapahtumatietojen perusteella**
2) **Vastaukset keskusvarastosta ja paikallisvarastoista tehtäviin kyselyihin (01)**
3) **Tilausten järjestely (P1, P2, P3) ostot mukaanluettuina**
4) **Viikottaiset varastoraportit, jotka sisältävät tietoja eri artikkelien kiertonopeudesta, myöhästyneistä toimituksista jne.**

Kuva 104. Varastonvalvontasysteemin kaavio

Järjestelmäkaavio

Järjestelmäkaavio kuvaa prosessia, jossa tapahtuu alkutietojen muunto jonkinlaisiksi tulostiedoiksi. Järjestelmäkaaviossa painopiste on lähinnä prosessin työvaiheissa ja niihin liittyvissä laitteissa ja välineissä. Ohjelmakaavio taas käsittelee pienempää kokonaisuutta, tietokoneen sisässä tapahtuvaa käsittelyä, kaavio on looginen esitys probleeman ratkaisusta ja näin ollen ohjelmoinnin seuraavan työvaiheen, koodauksen perusta.

Koska ohjelmakaaviossa on itse asiassa otettu jokin järjestelmäkaavion tietty osa tarkasteltavaksi, sisältyy siihen myös enemmän yksityiskohtia. Lisäksi ohjelmankin kaavioesitys voi koostua karkeasta runkokaaviosta, johon liittyvät osakaaviot puolestaan voivat ulottua hyvinkin pieniin detaljeihin.

Järjestelmäkaaviot ovat yleensä yksinkertaisempia ja helpompia laatia kuin ohjelmakaaviot. Niitä varten on käytettävissä myös enemmän symboleja ja enemmän liikkumisalaa. Kuvassa 104 on esitetty tyypillinen järjestelmäkaavio - esimerkki teknisessä mielessä ja symbolien käytön suhteen.

Ohjelmakaavio

Eräitä ohjelmakaavioihin liittyviä seikkoja on syytä erityisesti korostaa, koska ne niin olennaisesti ja läheisesti kuuluvat sekä ohjelmoinnin suunnitteluettä varsinaiseen toteutusvaiheeseen. Ohjelmoijan kannalta kaavio on eräänlainen yleistyökalu. Se on ohjelman rakennuspiirustus. Ohjelman suunnitteluvaiheessa lohkokaavio on mukana kaikessa, probleemasta käytettävissä olevien tietojen järjestelyssä, ratkaisujen hahmottelussa, ohjelmakokonaisuuden selvittelyssä jne. Lohkokaavio on itse asiassa eräs systemaattinen tapa tehdä muistiinpanoja.

Ohjelman suunnitteluvaiheessa voidaan lohkokaavion avulla 'testata' eri ratkaisuvaihtoehtoja - ja löytää loogisesti moitteeton ratkaisu. Kaavio hahmottuu ensin suuria toimintakokonaisuuksia kuvaavien symbolien muodostamaksi loogiseksi rungoksi. Tähän sisältyvät syöttö- ja tulostustoiminnat, toimintahaarat eri tietueiden käsittelyä varten jne.

Kun ohjelman looginen runko on selvillä, voidaan siitä edelleen erottaa lohkoja, jotka on tarpeen kuvata omina, runkokaavioon liittyvinä yksityiskohtaisina osakaavioina. Tehtävä muistuttaa kartan piirtämistä - ensin laaditaan laaja yleiskartta, joka sitten paloitellaan pienempiin ja yksityiskohtaisempiin osiin. Tällaista tekniikkaa nimitetään modulaariseksi. Se miten pitkälle yksityiskohtaiset esitykset viedään, riippuu probleeman laadusta ja laajuudestakin.

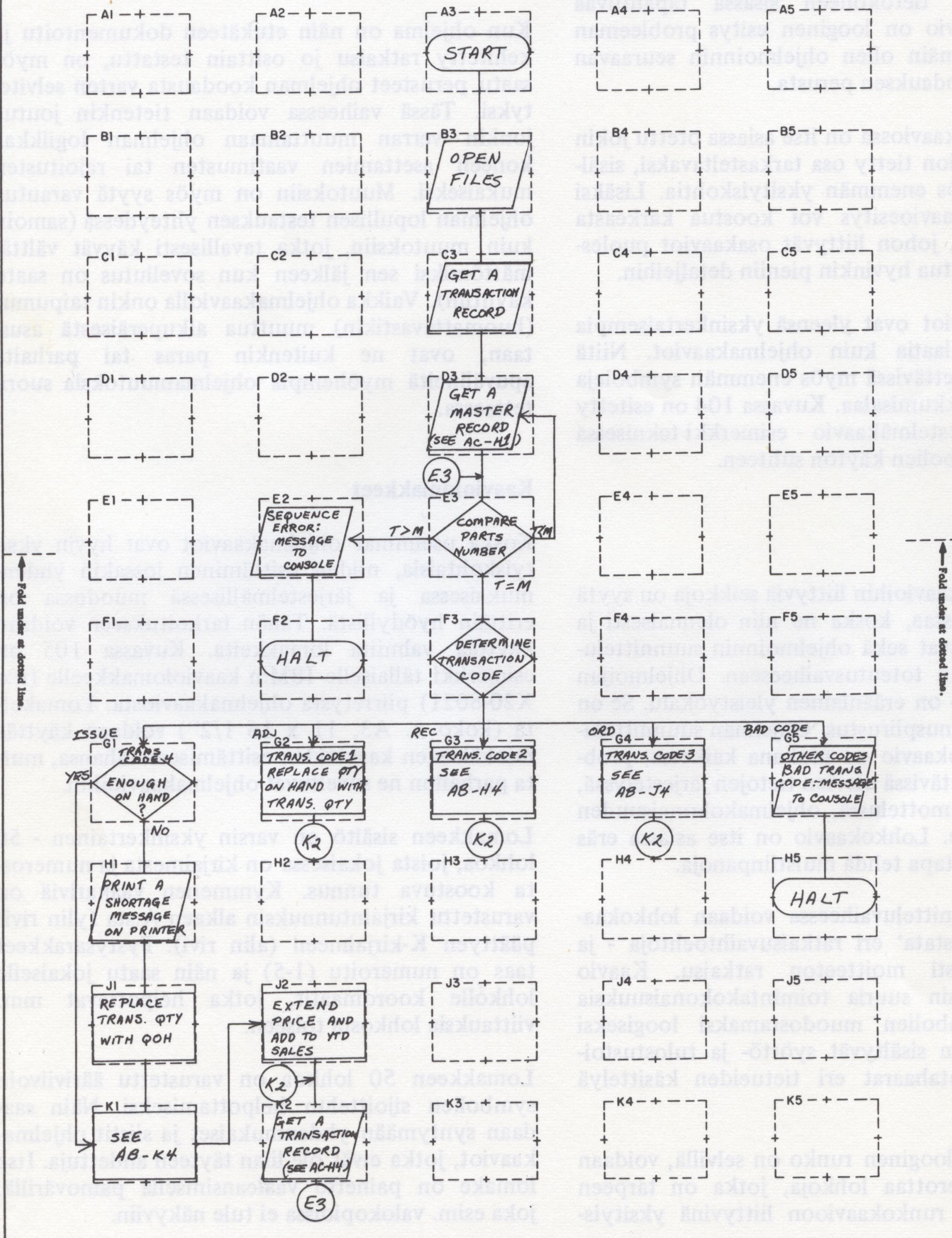
Kun ohjelma on näin etukäteen dokumentoitu ja kehitetty ratkaisu jo osittain testattu, on myös saatu perusteet ohjelman koodausta varten selvitetyksi. Tässä vaiheessa voidaan tietenkin joutua jonkin verran muuttamaan ohjelman logiikkaa koneen asettamien vaatimusten tai rajoitusten mukaiseksi. Muutoksiin on myös syytä varautua ohjelman lopullisen testauksen yhteydessä (samoin kuin muutoksiin, jotka tavallisesti käyvät välttämättömiksi sen jälkeen kun sovellutus on saatu käyntiin). Vaikka ohjelmakaaviolla onkin taipumus (huomattavastikin) muuttua alkuperäisestä asustaan, ovat ne kuitenkin paras tai parhaita apuvälineitä myöhempiä ohjelmamuutoksia suoritettaessa.

Kaaviolomakkeet

Koska useimmat ohjelmakaaviot ovat hyvin yksityiskohtaisia, niiden esittäminen jossakin yhdenmukaisessa ja järjestelmällisessä muodossa on erittäin hyödyllistä. Tähän tarkoitukseen voidaan käyttää valmiita lomakkeita. Kuvassa 105 on esimerkki tällaiselle IBM:n kaaviolomakkeelle (n:o X20-8021) piirretystä ohjelmakaaviosta. Lomaketta (koko n. A3, 11 x 16 1/2") voidaan käyttää minkälaisten kaavioiden esittämiseen tahansa, mutta parhaiten ne soveltuvat ohjelmakaavioihin.

Lomakkeen sisältö on varsin yksinkertainen - 50 lohkoa, joista jokaisessa on kirjaimesta ja numerosta koostuva tunnus. Kymmenen vaakariviä on varustettu kirjaintunnuksin alkaen A:sta (ylin rivi) päättyen K-kirjaimeen (alin rivi). Pystysarakkeet taas on numeroitu (1-5) ja näin saatu jokaiselle lohkolle koordinaatit, jotka helpottavat mm. viittauksia lohkosta toiseen.

Lomakkeen 50 lohkoa on varustettu ääriviivoin symbolien sijoittelun helpottamiseksi. Näin saadaan syntymään yhdenmukaiset ja siistit ohjelmakaaviot, jotka eivät ole liian täyteen ahdettuja. Itse lomake on painettu vaaleansinisellä painovärillä, joka esim. valokopioissa ei tule näkyviin.

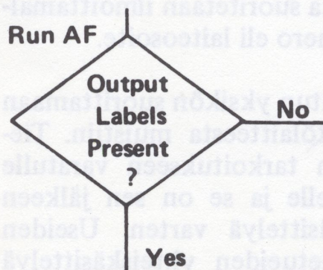


Kuva 105. Osa varastotilanteen päivitysohjelman lohkokaaviota

Lohkokaaviotekniikkaa

Seuraavassa on eräitä esimerkkejä lohkokaavioiden piirrostekniikasta:

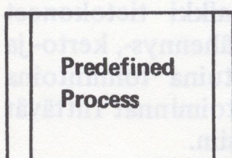
Ohjelman ja lohkokaavion väliset keskinäiset viittaukset. Käsky voidaan paikallistaa joko nimen perusteella taikka ko. koodauslomakkeen numeron ja käskyn rivinumeron perusteella. Viite voidaan kaaviossa sijoittaa symbolin vasempaan yläkulmaan (kuva 106).



Kuva 106. Esimerkki lohkokaavion viitejärjestelmästä

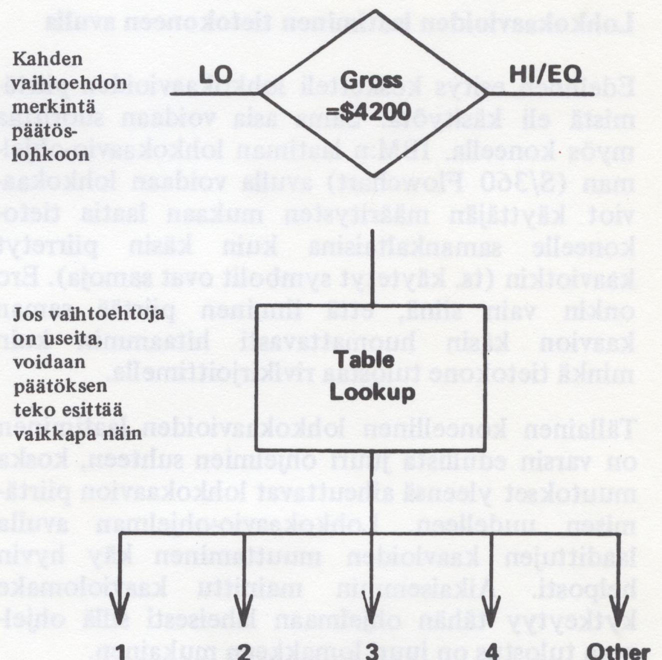
Yhdistelmäsymbolit muodostavat oman ryhmänsä. Vaikka ohjekortissa onkin kuvattu yli kolmekymmentä symbolia, mallilevy sisältää niitä noin kolmanneksen vähemmän. Nämä ylimääräiset symbolit saadaan aikaan yhdistelemällä mallilevyllä kuvattuja symboleja. Ohjekortissa yhdistelmäsymbolit on merkitty tähdellä. Esim. yhdistely (kahden tai useamman tiedoston) merkitään kärjellään seisovalla kolmiolla, samasta symbolista saadaan arkistoa kuvaava lisäämällä siihen vaakasuora viiva.

Loogiset ohjelmakokonaisuudet, itsenäiset rutiinit tms. voidaan ohjelman pääkaaviossa esittää kuvan 107 mukaisesti. Tällainen symboli merkitsee yleensä sitä, että rutiinista on toisaalla esitetty yksityiskohtainen lohkokaavio.



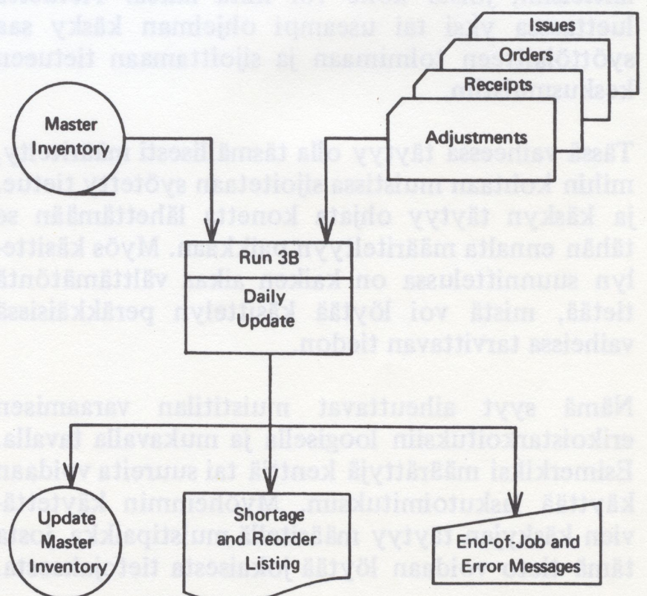
Kuva 107. Lohkojen merkintä

Päätöksentekoa, vertailua tms. kahden tai useamman vaihtoehdon välillä voidaan kuvata monin tavoin. Pari esimerkkiä on esitetty kuvassa 108. Näissä päätöslohkoissa tehty valinta ratkaisee mikä toiminto seuraavaksi suoritetaan, mihin rutiiniin haarautuminen tapahtuu tms.



Kuva 108. Esimerkkejä päätöslohkojen käytöstä

Kuvassa 105 esitettiin tyypillinen esimerkki ohjelmakaaviosta. Ohjelmakaavio (esimerkissä varastotietojen päivitys) perustuu järjestelmäkaavioon. Järjestelmäkaavio selvittää mistä päivittäinen ajo koostuu - syöttötiedostot, tapahtumat, joiden perusteella päivitys tapahtuu, tulostiedostot jne. (kuva 109).



Kuva 109. Systeemikaavio

Lohkokaavioiden laatiminen tietokoneen avulla

Edellinen esitys kosketteli lohkokaavioiden piirtämistä eli käsityötä. Sama asia voidaan suorittaa myös koneella. IBM:n laatiman lohkokaavio-ohjelman (S/360 Flowchart) avulla voidaan lohkokaaviot käyttäjän määritysten mukaan laatia tietokoneelle samankaltaisina kuin käsin piirretyt kaaviotkin (ts. käytetyt symbolit ovat samoja). Ero onkin vain siinä, että ihminen piirtää saman kaavion käsin huomattavasti hitaammin kuin minkä tietokone tulostaa rivikirjoittimella.

Tällainen koneellinen lohkokaavioiden laatiminen on varsin edullista juuri ohjelmien suhteen, koska muutokset yleensä aiheuttavat lohkokaavion piirtämisen uudelleen. Lohkokaavio-ohjelman avulla laadittujen kaavioiden muuttaminen käy hyvin helposti. Aikaisemmin mainittu kaaviolomake kytkeytyy tähän ohjelmaan läheisesti sillä ohjelman tulostus on juuri lomakkeen mukainen.

TIETOJEN LUKU

Syöttölaitteen täytyy ensin lukea kaikki tietojenkäsittelyjärjestelmään tulevat tiedot ja siirtää ne sitten keskusmuistiin. Jokaiselle syöttöpaikalle on annettu oma osoite samalla tavalla kuin muistipaikoillekin.

Tietojenkäsittelytehtävään liittyy tavallisesti kokonaisia tiedostoja. Nämä tiedostot sijoitetaan syöttölaitteisiin, joista kone voi niitä lukea. Tietuetta luettaessa yksi tai useampi ohjelman käsky saa syöttölaitteen toimimaan ja sijoittamaan tietueen keskusmuistiin.

Tässä vaiheessa täytyy olla täsmällisesti määritelty, mihin kohtaan muistissa sijoitetaan syötetty tietue, ja käskyn täytyy ohjata konetta lähettämään se tähän ennalta määriteltyyn paikkaan. Myös käsittelyn suunnittelussa on kaiken aikaa välttämätöntä tietää, mistä voi löytää käsittelyn peräkkäisissä vaiheissa tarvittavan tiedon.

Nämä syyt aiheuttavat muistitilan varaamisen erikoistarkoituksiin loogisella ja mukavalla tavalla. Esimerkiksi määrättyjä kenttiä tai suureita voidaan käyttää laskutoimituksiin. Myöhemmin käytettävien käskyjen täytyy määritellä muistipaikka, josta tämä tieto voidaan löytää jokaisesta tietojaksosta.

Tietokonejärjestelmissä, joihin kuuluu käyttöjärjestelmä, vapautuu ohjelmoija huolehtimasta varsinaisesta syöttö- ja tulostustoiminnasta - hänen tehtäväkseen jää loogisten tietueiden käsittely, jotka käyttöjärjestelmä toimittaa ohjelman käytettäväksi.

Lukutoiminta käsittää seuraavat eri toiminnat:

1. Valitaan syöttölaite ja saatetaan se toimintavalmiiksi ennen varsinaisen lukemisen alkamista. Valittu laite on se laite, jolla päästään käsiksi ohjelmoijan määräämiin asiaankuuluviin tiedostoihin. Laitteen valinta suoritetaan ilmoittamalla määrätty koodinumero eli laiteosoite.

2. Lukukäsky panee valitun yksikön suorittamaan tietueen siirron syöttölaitteesta muistiin. Tietue sijoitetaan tähän tarkoitukseen varatulle erityiselle muistialueelle ja se on sen jälkeen käytettävissä jatkokäsittelyä varten. Useiden toisiinsa liittyvien tietueiden yhteiskäsittelyä varten voidaan varata useita lukualueita. Tällaisia tietueita ovat esimerkiksi päätietojakso ja sen vastaavat yksityiset tapahtumat.

3. Lukukäskyjen järjestys ohjelmassa määrää tiedostojen lukemisjärjestyksen. Toiset käskyt vertaavat myöhemmin eri tiedostoista luettuja tietojaksoja määrätäkseen yksityiskohtien suhteen tietueeseen, yksityiskohtien keskinäiset suhteet jne.

4. Yhdellä kertaa muistiin sijoitettavien tietueiden määrä riippuu tiedostojen rakenteesta, käsiteltävien tietueiden tyypistä ja pituudesta sekä käytettävissä olevasta muistikapasiteetista.

LASKEMINEN

Niin pian kuin tiedot on luettu tietojenkäsittelyjärjestelmään ja sijoitettu määrättyihin muistipaikkoihin, voi laskeminen alkaa. Kaikki tietokoneet pystyvät suorittamaan yhteen-, vähennys-, kerto- ja jakolaskua joko sisääntallennettuina toimintoina tai ohjelman ohjaamana. Nämä toiminnat riittävät useimpiin kaupallisiin sovellutuksiin.

Myös useissa kehittyneemmissäkin tieteellisissä laskentatehtävissä voidaan mitä monimutkaisimmat yhtälöt pelkistää alkeisaritmeettisiksi toimintavaiheiksi. Kuitenkin jotkut järjestelmät voivat suorittaa monia erikoistoimintoja, jolloin matemaattisten probleemien ratkaisu helpottuu.

Jokaiseen yksinkertaiseen laskutoimitukseen tarvi-

taan ainakin kaksi tekijää: kertoja ja kerrottava, jakaja ja jaettava jne. Koneen aritmeettinen yksikkö operoi näillä tekijöillä saadakseen aikaan tuloksen, esimerkiksi tulon tai osamäärän. Jokaisessa laskutoimituksessa tarvitaan sen tähden ainakin kaksi muistipaikkaa. Toinen suure on tavallisesti keskusmuistissa ja toinen rekisterissä. Systeemissä/360 molemmat suureet voivat sijaita rekisterissä.

Laskutoimitus voidaan aloittaa sijoittamalla toinen tekijä rekisteriin ja tyhjentämällä se samalla kertaa kaikista aikaisemmista tekijöistä tai tuloksista, joita se saattaa sisältää. Käskyn osoiteosa ilmoittaa ensimmäisen tekijän muistipaikan, toiminta itse sisältää rekisterin tai muun muistiyksikön osoitteen. Muutamissa koneissa on laskutoimituksia varten käytettävissä useampia kuin yksi rekisteri. Tässä tapauksessa osoitteen on ilmoitettava myös, mitä rekisteriä käytetään.

Kun toinen tekijöistä on asianmukaisesti sijoitettu laskulaitteeseen tai muuhun sopivaan rekisteriin, varsinainen laskutoimitus toimeenpannaan käskyllä, jonka operaatio-osa määrittää suoritettavan aritmeettisen toiminnan ja jonka operandina on toisen tekijän osoite. Tietokone operoi näillä kahdella tekijällä, joista toinen on laskulaitteessa ja toinen muistissa, ja sijoittaa tuloksen laskulaitteeseen.

Tulos palautetaan keskusmuistiin toisella käskyllä, joka tallentaa kentän johonkin tuloksen sijoittamista varten määrättyyn muistipaikkaan. Kenttä on ryhmä toisiinsa liittyviä kirjaimia tai numeroita, jotka ilmaisevat suuretta, määrää, nimeä, yhtäläisyyttä jne.

Yksinkertaisten käskysarjojen avulla voidaan suorittaa kaikkia käytännöllisiä laskutoimituksia, myös sellaisia, joissa käytetään useampia tekijöitä. Toisin sanoen tekijä voidaan sijoittaa laskulaitteeseen ja kertoa se jollakin luvulla ja tuloon voidaan lisätä tai vähentää siitä useita muita tekijöitä. Siten voidaan suorittaa esimerkiksi jakolasku ja saatua osamäärää hyväksikäyttäen voidaan suorittaa taas uusia yhteen- ja vähennyslaskuja. Välitulokset voidaan tallentaa joka kerta.

Rekisteriin voidaan sijoittaa esimerkiksi työtunnit sisältävä kenttä ja kertoa se tuntipalkalla, jolloin saadaan ansio. Urakkatyö- ja voitto-osuussummat voidaan sen jälkeen lisätä ansioon, jolloin saadaan varsinaisten ansioiden kokonaissumma, joka tallennetaan palkkatietojaksoon. Varsinainen ansio voidaan sen jälkeen jakaa tuntimäärällä, niin että saadaan keskimääräinen tuntiansio. Tämä tuntiansio kerrotaan 1,5-kertaisella ylityötuntien määrällä, jolloin saadaan ylityöansioiden markkamääräinen summa. Sitten lasketaan kokonaisbruttoansio ja tallennetaan se. Vero lasketaan bruttosummaa käyttäen, muut palkkatiedot kootaan eri tavoilla lasketuista summista. Vero- ja vähennysvälitulokset sekä lopuksi maksettava nettosumma tallennetaan kaikki palkkatietojaksoon.

Rekisterin sisältöä siirtävät ja pyöristävät toiminnat on tarkoitettu tulosten järjestelemiseen, pidentämiseen tai lyhentämiseen. Näillä toiminnoilla voidaan käsitellä desimaalilukuja ja koneelle voidaan antaa ohjeet desimaalipilkun sijoittamisesta.

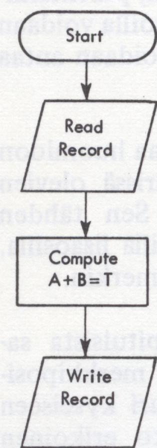
Kaikissa laskutoimituksissa täytyy ottaa huomioon muistissa tai asiaankuuluvassa rekisterissä olevien tekijöiden algebrallinen etumerkki. Sen tähden tietokonejärjestelmä on varustettu eräillä lisäosilla, jotka tunnistavat ja tallentavat tekijän merkin.

Jos tietojaksot muodostuvat kiinteäpituisista sanoista, yksi sanan positio määrätään merkkipositioksi ja se liittyy automaattisesti juuri kyseiseen sanaan. Laskulaitteissa on myös joko erikoinen muistin merkkipositio tai merkki-indikaattori, joka on ohjelmoijan käytettävissä. Tällä tavoin tuloksen etumerkki määrätään varsinaisen laskutoimituksen kanssa samanaikaisesti. Tietokone noudattaa algebran sääntöjä kaikissa aritmeettisissa peruslaskutoimituksissa.

Sanojen, suureiden ja arvojen pituus riippuu käytetyn järjestelmän rakenteesta. Tekijöiden sijoituksen, tuloksen pituuden jne. määräävät täsmälliset säännöt vaihtelevat jonkin verran eri järjestelmissä. Kaikissa niissä tapauksissa, joissa tuloksen arvioidaan ylittävän laskulaitteen tai muistirekisterin kapasiteetin, ohjelmoijan on järjestettävä tieto siten, että ensin muodostuu osatuloksia ja nämä osatulokset sitten yhdistetään lopputulokseksi. Joitakin lyhennystoimintoja suoritetaan sen vuoksi, että voitaisiin käsitellä mukavasti hyvin suuria tai pieniä lukuja ja murtolukuja. Alunperin matemaattisiin sovellutuksiin suunnitellut koneet sisältävät tätä tarkoitusta varten yleensä joukon aritmeettisia erikoistoimintoja (ks. 'Liukuvan pilkun aritmetiikka').

Laskutoimitukset suoritetaan kaikissa tietojenkäsittelyjärjestelmissä huomattavasti suuremmilla nopeuksilla kuin syöttö- tai tulostustoiminnat, koska lukeminen ja kirjoittaminen vaativat mekaanisten

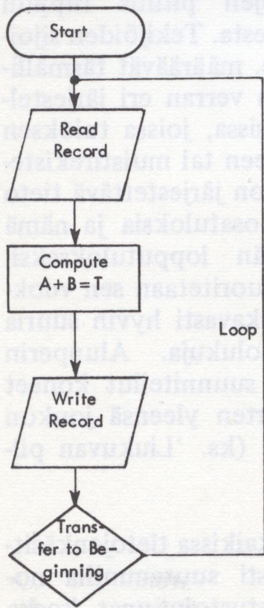
laitteiden käyttämistä ja tietomateriaalin liikkumista, kun taas laskutoimitukset suoritetaan sähköisesti. Monissa kaupallisissa sovellutuksissa ovat laskutoimitukset suhteellisen yksinkertaisia ja järjestelmän kokonaisnopeuden määrää tavallisesti syöttö- ja tulostusyksiköiden nopeus. Matemaattisissa sovellutuksissa tilanne on päinvastainen, laskutoimitukset ovat monimutkaisia ja suuret laskunopeudet ovat välttämättömiä. Jokaisen järjestelmän suunnittelussa täytyy saavuttaa realistinen tasapaino laskukyvyn ja tietojaksojen käsittelykyvyn välillä.



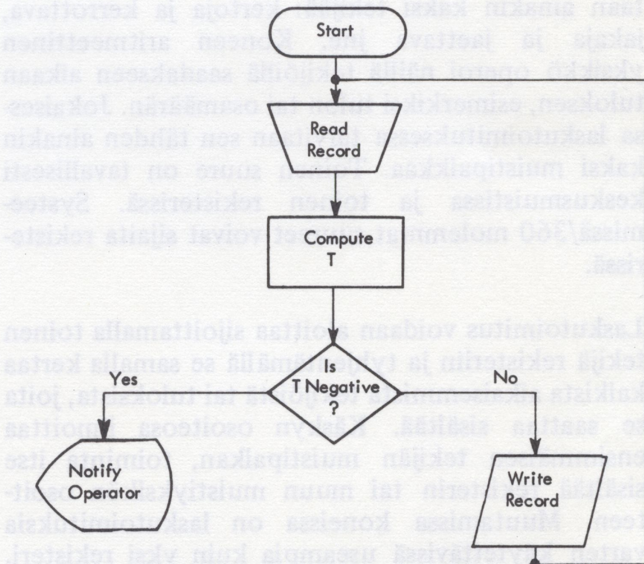
Kuva 110. Ohjelmakaavio A+B = T

LOOGISET TOIMINNAT

Tietokoneen noudattama käskyjen toimeenpanojärjestys määräytyy siten, että kone voi joko ottaa

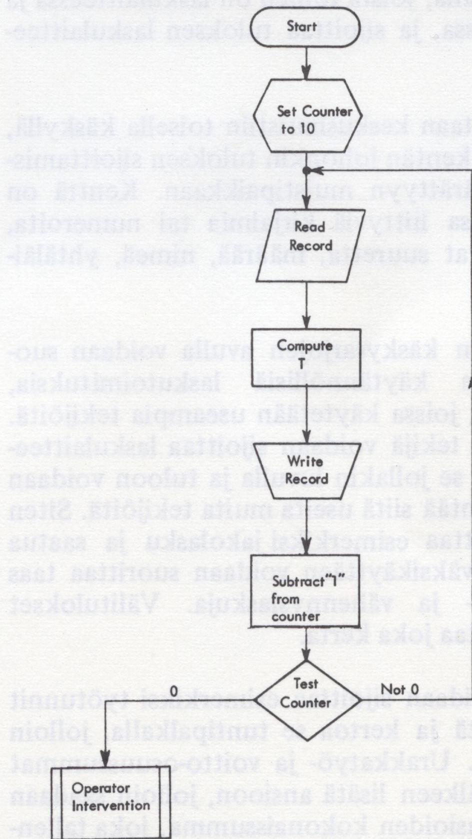


Kuva 111. Ohjelmasilmukka



Kuva 112. Ehdollinen hyppy

käskyt peräkkäisistä muistipaikoista tai jonkin käskyn operandiosa osoittaa seuraavan suoritettavan käskyn muistipaikan. Jos käskyt olisi toimeenpantava aina niiden sijaintipaikkojen mukaisessa peräkkäisjärjestyksessä, ohjelmalla olisi käytettävis-



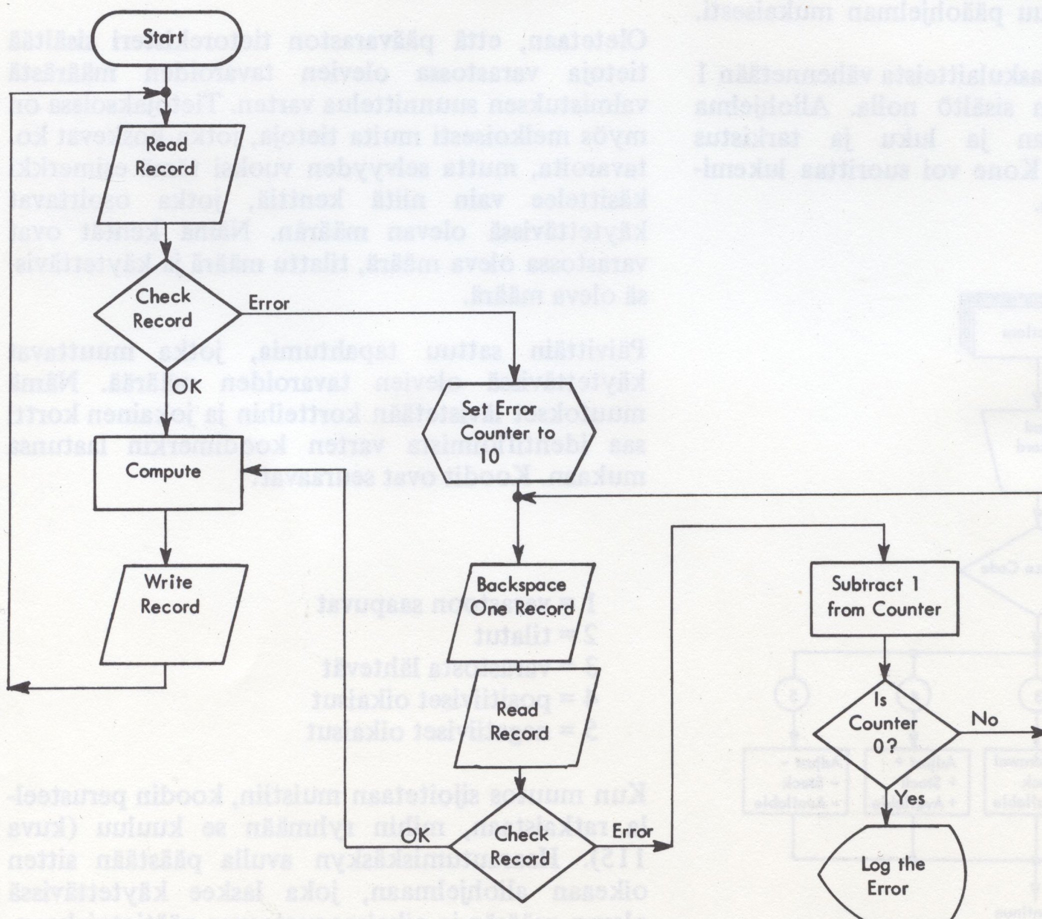
Kuva 113. Tietueiden lukumäärään perustuva ehdollinen hyppy

sään vain yksi ainoa toimintajärjestys, eikä sillä olisi mahdollisuutta ottaa huomioon poikkeustapauksia tai määrättyjen, ohjelman toimeenpanon aikana esiintyvien edellytysten perusteella valita oikeaa vaihtoehtoa. Ja lisäksi, koska ei olisi mahdollista saada konetta toistamaan tiettyä käskysarjaa, olisi välttämätöntä laatia täydellinen ohjelma jokaista tietojaksoa varten.

Tarkastellaanpa ohjelmaa, jonka lohkokaavio on esitetty kuvassa 110. Tämä ohjelma yksinään laskee T:n vain yhdelle tietojaksolle. Palaamalla takaisin ensimmäiseen käskyyn voidaan kuitenkin käydä läpi mielivaltainen määrä jaksoja toistamalla sama ohjelma silmukkana. Tässä tarkoituksessa annetaan vielä yksi käsky, joka palauttaa tietokoneen takaisin ensimmäiseen käskyyn (kuva 111). Kun tämä ohjelma on kerran alkanut, se jatkuu, kunnes ei enää ole käsiteltäviä jaksoja. Ohjelmasilmukat ovat yleisesti käytössä, ja niistä voidaan päästä ulos monella tavalla.

Tietokonetta voidaan esimerkiksi käskeä testaamaan T joka kerta kun se on laskettu ja pysähtymään T:n arvon tullessa negatiiviseksi (kuva 112). Tässä tapauksessa on kysymyksessä ehdollinen haarautuminen. Ohjelma toistetaan vain, jos jokin aikaisemmin asetettu ehto on täytetty. Tietokone voidaan myös käskeä suorittamaan ohjelma kymmenellä jaksolla ja sitten pysähtymään (kuva 113). Oletetaan, että vakiot 10 ja 1 ovat tietokoneessa ja että 10:stä vähennetään 1 joka kerta, kun silmukka on kierretty. Kymmenen kierroksen jälkeen on nolla siinä muistipaikassa, jossa aikaisemmin oli 10. Siirto- eli haarautumiskäsky johtaa ulos silmukasta.

Ehdollista haarautumiskäskyä voidaan käyttää siirryttäessä erikoistarkoitukseen varattuun aliohjelmaan, jolloin aliohjelman toimeenpano voidaan suorittaa varsinaisen ohjelman ulkopuolella. Tämä aliohjelma suoritetaan vain silloin, kun tietokone on todennut edeltäkäsin määrätyn ehdon täyttymisen.



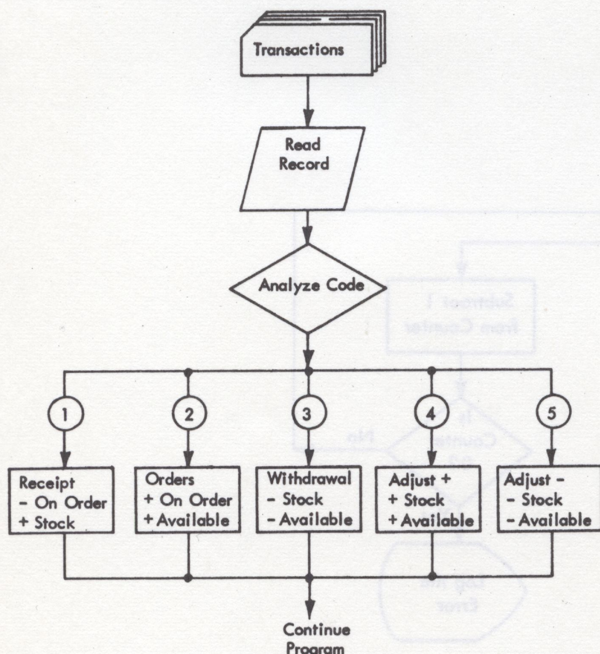
Kuva 114. Lukuvirheen käsittelyrutiini

misen tai poikkeustilanteen.

Tyypillinen esimerkki aliohjelman käytöstä on virheiden hakeminen tietojaksoista, jotka luetaan magneettinauhalta tai kirjoitetaan nauhalle. Kun kukin tietojakso saapuu keskusyksikköön tai lähtee siitä, tutkitaan luku- tai kirjoitusvirheen indikaattoria. Jos indikaattori on virittyneenä, tietokoneen käsketään siirtyä aliohjelmaan, joka sisältää käskyt, joiden mukaan virhe yritetään oikaista. Kuvassa 114 esitetyssä lohkokaaviossa on ohjelmasilmukka mahdollisesti esiintyviä lukuvirheitä varten. Samanlainen silmukka voitaisiin laatia myös kirjoituksen tarkistusta varten.

Jos lukuvirhe löytyy, ohjelma haarautuu virhealiohjelmaan. Laskijaan asetetaan luku 10, jotta voidaan laskea, kuinka monta kertaa virheellistä jaksoa yritetään lukea. Nauha siirretään takaisin virheellisesti luetun jakson yli ja luetaan se uudestaan. Suoritetaan uusi tarkistus ratkaisemaan, onko tämä luku tapahtunut oikein. Jos se on, haarautumiskäsky palauttaa toiminnan pääohjelmaan ja käsittely jatkuu pääohjelman mukaisesti.

Jos lukuvirhe toistuu, laskulaitteista vähennetään 1 ja testataan, onko sen sisältö nolla. Aliohjelma toistetaan vielä kerran ja luku ja tarkistus suoritetaan uudelleen. Kone voi suorittaa lukemisen kaikkiaan 10 kertaa.



Kuva 115. Koodiin perustuva hyppy

Ellei virhettä voida korjata, siirretään toimintaoikeus rutiinille, joka rekisteröi virheellisen tietueen tai lohkon ja sen jälkeen palataan lukemaan seuraavaa tietuetta tai lohkoa. 2400-sarjan 9-uraisten nauhojen tarkistusjärjestelmä korjaa nauhan jollakin uralla tapahtuneen virheen automaattisesti, joten tämän tyyppisiä virhetilanteita ei ohjelmassa tarvitse huomioida. Yleensäkin voidaan sanoa, että kaikkinaisten virhetilanteiden tarkistus- ja korjaustoiminnot kuuluvat käyttöjärjestelmälle (IBM:n toimittamalle ohjelmointijärjestelmälle, joita on useimpia tietokonejärjestelmiä varten olemassa) eikä niitä niin muodoin tarvitse sisällyttää varsinaisiin käyttöohjelmiin.

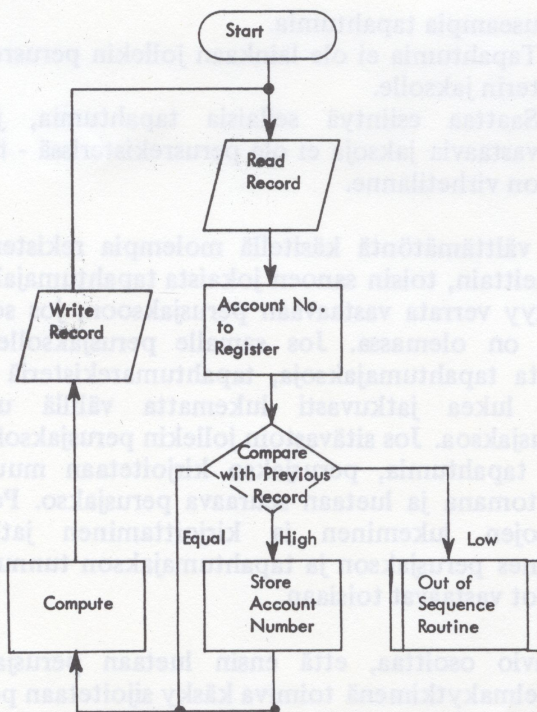
Ohjelma voidaan laatia myös siten, että kone voi erottaa yhden tai useampia tietojaksotyyppejä yhdestä ainoasta tietorekisteristä. Kulloinkin valitaan tietojaksotyypin mukaan määrätty ohjelman haara. Tällaista menettelyä käytetään varsinkin silloin, kun päärekisteristä haetaan tapahtumajaksoja vastaavia pääjaksoja (esimerkiksi rekisterin kunnossapitosovellutus).

Oletetaan, että päävaraston tietorekisteri sisältää tietoja varastossa olevien tavaroiden määrästä valmistuksen suunnittelua varten. Tietojaksoissa on myös melkoisesti muita tietoja, jotka koskevat ko. tavaroita, mutta selvyyden vuoksi tämä esimerkki käsittelee vain niitä kenttiä, jotka osoittavat käytettävissä olevan määrän. Nämä kentät ovat varastossa oleva määrä, tilattu määrä ja käytettävissä oleva määrä.

Päivittäin sattuu tapahtumia, jotka muuttavat käytettävissä olevien tavaroiden määrää. Nämä muutokset lävistetään kortteihin ja jokainen kortti saa identifioimista varten koodimerkin laatunsa mukaan. Koodit ovat seuraavat:

1 = varastoon saapuvat
2 = tilatut
3 = varastosta lähtevät
4 = positiiviset oikaisut
5 = negatiiviset oikaisut

Kun muutos sijoitetaan muistiin, koodin perusteella ratkaistaan, mihin ryhmään se kuuluu (kuva 115). Haarautumiskäskyn avulla päästään sitten oikeaan aliohjelmaan, joka laskee käytettävissä olevan määrän ja oikaisee vastaavan päätietojakson. Päätietojakson lukeminen ja kirjoittaminen ei käy ilmi kaaviosta.

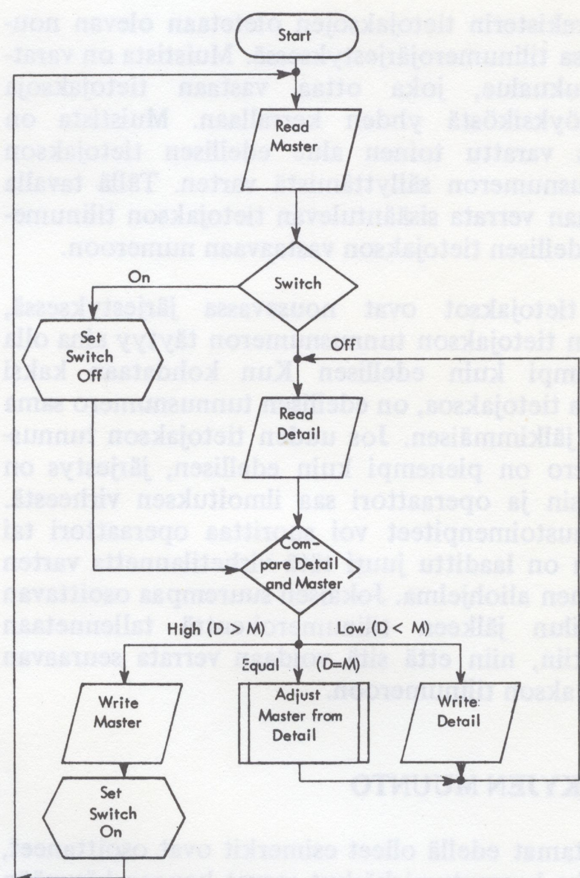


Kuva 116. Järjestyksen tarkistus

VERTAILU

Vertailutoiminnat laajentavat oleellisesti tietokoneen kykyä tehdä itse ratkaisuja rajoitetussa määrässä ohjelman logiikan perusteella. Sellaisten toimintojen avulla tietokone voi todeta, onko kaksi muistissa olevaa tietokenttää samanarvoisia vai onko toinen pienempi tai suurempi kuin toinen. Vertailun peruste on etukäteen määrätyn, koneen virtapiireihin sisältyvän arvojärjestyksen mukainen. Tämä järjestys voi olla esimerkiksi tavallinen nouseva järjestys, jolla tietojaksot erotetaan toisistaan. Esimerkiksi tavallisessa lukujonossa 0-9 otetaan luku 9 jonon korkeimmaksi merkiksi. Samalla tavalla oletetaan kirjaimen Z olevan aakkosten korkein kirjain. Samasta syystä on tietokoneelle luku 162 suurempi kuin luku 159 ja nimi JONES pienempi kuin SMITH. Erikoismerkkejä, kuten , ,+ ja -, voidaan myös sisällyttää tähän arvojärjestykseen, koska näitä merkkejä täytyy käsitellä tietoina raportteja kirjoitettaessa ja monissa erikoistarkoituksissa.

Vertailutoimintoja käytetään tietorekistereiden järjestyksen tarkistusohjelmissa, lajittelussa jne. Vertailu tietojaksossa olevan tunnuskentän ja toisessa tietojaksossa olevan samanlaisen kentän välillä tekee koneelle mahdolliseksi käsitellä joukkoa



Kuva 117. Ohjelmakytkin

samaan alaan kuuluvia tietorekistereitä samanaikaisesti edellyttäen, että kaikki tietorekisterit on järjestetty joko nousevaan tai laskevaan järjestykseen tämän yhteisen kentän mukaan.

Toinen (tai kumpikin) kenttä sijoitetaan rekisteriin. Vertailukäsky suorittaa sen jälkeen vertailun rekisterissä olevan kentän ja toisen, muistissa sijaitsevan kentän välillä. (Systeemissä/360 vertailtavat kentät voivat sijaita molemmat rekisterissä, toinen rekisterissä ja toinen muistissa taikka molemmat muistissa). Käskyn toteuttamisen jälkeen vertailun tulos - yhtäsuuri, suurempi tai pienempi on ohjelmoijan käytettävissä. Toisin sanoen ohjelmoija voi testata minkälainen vertailun tulos oli, tutkia mikä em. kolmesta 'indikaattorista' on asettunut päälle ja tuloksen mukaan haarautua tarvittavaan rutiiniin tai jatkaa seuraavasta käskystä.

Kuvassa 116 esitetään tyypillinen yksinkertainen rekisterin järjestyksen tarkistava ohjelma. Kaikkien

tietorekisterin tietojaksojen oletetaan olevan nousevassa tilinumerojärjestyksessä. Muistista on varattu lukualue, joka ottaa vastaan tietojaksoja syöttöyksiköstä yhden kerrallaan. Muistista on myös varattu toinen alue edellisen tietojakson tunnusnumeron säilyttämistä varten. Tällä tavalla voidaan verrata sisääntulevan tietojakson tilinumeroa edellisen tietojakson vastaavaan numeroon.

Jos tietojaksot ovat nousevassa järjestyksessä, uuden tietojakson tunnusnumeron täytyy aina olla suurempi kuin edellisen Kun kohdataan kaksi samaa tietojaksoa, on edellisen tunnusnumero sama kuin jälkimmäisen. Jos uuden tietojakson tunnusnumero on pienempi kuin edellisen, järjestys on sekaisin ja operaattori saa ilmoituksen virheestä. Korjaustoimenpiteet voi suorittaa operaattori tai sitten on laadittu juuri tätä virhetilannetta varten erillinen aliohjelma. Jokaisen suurempaa osoittavan vertailun jälkeen tilinumerokenttä tallennetaan muistiin, niin että sitä voidaan verrata seuraavan tietojakson tilinumeroon.

KÄSKYJEN MUUNTO

Muutamat edellä olleet esimerkit ovat osoittaneet, kuinka haarautumiskäskyt saavat koneen käymään ohjelman läpi eri teitä kuin tavallisesti. Toimeenpantava ohjelmasilmukka riippuu aikaisemmin suoritetun vertailun tuloksesta tai tiettyjen indikaattoreiden asennoista. Indikaattorit voivat virittyä tuloksen ollessa nolla, virhetilanteissa jne.

Toinen menetelmä ohjelman kulun muuttamiseksi on itse käskyjen toimintaosan muuttaminen eli ns. modifioiminen. Käskyn modifiointia voidaan käyttää esimerkiksi asettamaan ohjelmakytkin (SWITCH), joka ohjaa koneen toiminnan toiseen kahdesta mahdollisesta ohjelmahaarasta. Kytkin viritetään tai poistetaan käskyllä. Kytkimen käyttö on esitetty kuvassa 117.

Olettakaamme, että luettavana on kaksi tietorekisteriä. Ne on molemmat järjestetty nousevaan järjestykseen yhteisen kentän mukaan. (Esimerkiksi osanumero, tilinumero, käyttönumero, jne.) Toinen rekisteri on perusrekisteri, toinen on tapahtumarekisteri, joka sisältää perusrekisteriä koskevia tietoja. Kun tapahtumajaksoja vastaavia perusrekisterin tietojaksoja oikaistaan päivän tasalle, voi esiintyä kolmenlaisia tilanteita:

1. Samalle perusrekisterin tietojaksolle on yksi tai useampia tapahtumia.
2. Tapahtumia ei ole lainkaan jollekin perusrekisterin jaksolle.
3. Saattaa esiintyä sellaisia tapahtumia, joita vastaavia jaksoja ei ole perusrekisterissä - tämä on virhetilanne.

On välttämätöntä käsitellä molempia rekistereitä vaiheittain, toisin sanoen jokaista tapahtumajaksoa täytyy verrata vastaavaan perusjaksoon, jos sellainen on olemassa. Jos samalle perusjaksolle on useita tapahtumajaksoja, tapahtumarekisteriä täytyy lukea jatkuvasti lukematta välillä uutta perusjaksoa. Jos sitävastoin jollekin perusjaksolle ei ole tapahtumia, perusjakso kirjoitetaan muuttumattomana ja luetaan seuraava perusjakso. Perusjaksojen lukeminen ja kirjoittaminen jatkuu, kunnes perusjakson ja tapahtumajakson tunnusnumerot vastaavat toisiaan.

Kaavio osoittaa, että ensin luetaan perusjakso. Ohjelmakytkimenä toimiva käsky sijoitetaan perusjakson lukemisen jälkeen, mutta ennen tapahtumajakson lukemista. Toiminnan alkaessa kytkin ei ole virittyneenä, joten tapahtuu tapahtumajakson lukeminen. Tapahtumajakson tunnusnumeroa verrataan perusjakson vastaavaan numeroon. Jos ne ovat yhtäsuuret, perusjaksoa korjataan tapahtumajakson sisältämillä muutoksilla ja luetaan seuraava tapahtumajakso. Jos tämä muutos ei koske viimeksi luettua perusjaksoa, joka on tallennettu muistiin, sen tunnuskentän pitäisi olla verrattaessa suurempi. Aikaisemmin korjattu perusjakso kirjoitetaan ja ohjelmakytkin viritetään. Uusi perusjakso sijoitetaan muistiin, mutta koska kytkin on virittyneenä, uutta tapahtumajaksoa ei lueta, vaan sensijaan toiminta siirtyy suoraan vertailukäskyyn. Kytkin poistetaan joka kerta tämän tapahtuessa. Toiminta jatkuu vertaamalla tällä tavalla jokaista uutta muistiin sijoitettua jaksoa. Jos tapahtumajakson tunnusnumero on pienempi kuin perusjakson, tapahtumajakso kirjoitetaan erillisellä tulosyksiköllä ja luetaan uusi tapahtuma.

Jos kytkin on virittyneenä, kytkimenä toimivan käskyn operaatio-osa merkitsee ehdotonta haarautumista. Osoiteosa ilmoittaa vertailukäskyn sijainnin. Kun kytkin poistetaan, operaatio-osa muutetaan sellaiseksi, että käsky luetaan, mutta ei suoriteta mitään toimintaa. Tässä tapauksessa kone jättää käskyn huomioimatta ja siirtyy seuraavaan käskyyn, joka aiheuttaa tapahtumajakson lukemisen.

OSOITTEEN MUUNTO

Käskyjen osoiteosia voidaan myös käsitellä tietoina. Käskyn osoitetta voidaan muuttaa aritmeettisesti, sitä voidaan haluttaessa verrata muihin osoitteisiin tai tekijöihin, tai sen sijaintipaikkaa muistissa voidaan muuttaa. Osoitemodifiointia käytetään kahdesta syystä:

1. Ohjelman käskyjen lukumäärää voidaan pienentää ja saada siten enemmän tilaa tiedoille ja muille tekijöille. Yksi käsky tai käskysarja voi ilmoittaa useiden muistipaikkojen osoitteet.

2. Ohjelman ohjaaman työn peruskaavio voi toimia mallina, joka voi muuttua riippuen syöttötiedoista, laskutoimintojen tuloksista, tietorekisterin loppumisesta jne.

Esimerkiksi käskyn toiminnan kohdistuessa taulukkoon on sen osoiteosa vaivattomasti muunnettavissa (rekisterin sisältöä muuttamalla). Osoitemodifioinnilla on tärkeä tehtävänsä myös esim. tietojen kaukokäsittelyohjelmissa, joissa muuntotekijänä voidaan käyttää linjanumeroa, terminaalin numeroa jne.

INDEKSOINTI

Monissa tietokoneissa voidaan käskyn osoiteosaa muuttaa lisäämällä siihen tai vähentämällä siitä erilaisia suureita, jotka sijaitsevat yhdessä tai useammassa erikoislaskulaitteessa. Laskulaitetta voidaan kutsua indeksirekisteriksi, jos se on erikoisesti tähän tarkoitukseen varattu, tai se voi olla edeltäkäsin määrätty paikka muistissa, ns. indeksisana. Tietokoneessa voi olla useita indeksirekistereitä tai useita muistipaikkoja indeksisanoja varten. Sekä indeksirekisterillä että indeksisanalla on sama tarkoitus, kuitenkin indeksisana on tavallisesti paremmin ohjelmaan soveltuva ja siten usein käyttökelpoisempi kuin indeksirekisteri.

Indeksointilaitteella varustetut tietokoneet käyttävät laajennettua käskymuotoa, niin että käskyn operandin osaksi voidaan määrittää yksityinen rekisteri tai sana.

Oletetaan, että 50 suuretta on sijoitettu muistipaikkoihin 1001-1050 ja nämä suureet on laskettava yhteen laskulaitteen sisällön kanssa. Ellei käytettäisi indeksointia eli osoitemodifiointia, ohjelmassa olisi pakko toistaa yhteenlaskukäsky 50 kertaa, jolloin jokaisen käskyn operandina oleva osoite olisi yhtä suurempi kuin edellisen, esimerkiksi ADD 1001, ADD 1002, ADD 1003 jne.

Indeksoinnin avulla tullaan toimeen yhdellä yhteenlaskukäskyllä, ADD 1051, jonka operandista vähennetään indeksirekisterin sisältö.

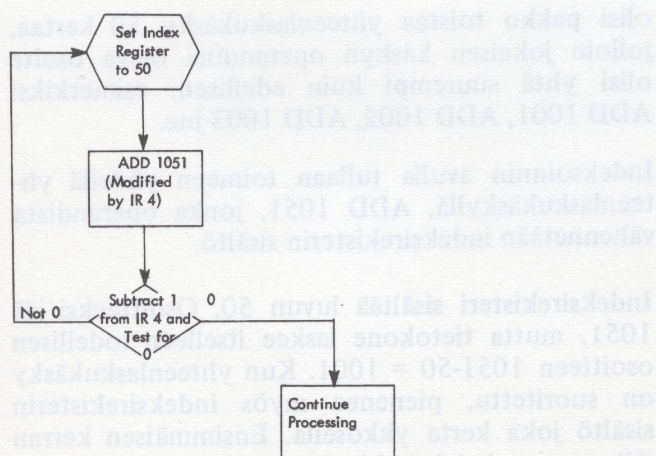
Indeksirekisteri sisältää luvun 50. Osoitteeksi jää 1051, mutta tietokone laskee itselleen todellisen osoitteen 1051-50 = 1001. Kun yhteenlaskukäsky on suoritettu, pienenee myös indeksirekisterin sisältö joka kerta ykkösellä. Ensimmäisen kerran jälkeen on indeksirekisterissä siis 49. Kun sama yhteenlaskukäsky toistetaan ja operandista vähennetään indeksirekisterin sisältö, saadaan todelliseksi osoitteeksi 1051-49 = 1002. Ohjelmasilmukan avulla voidaan yhteenlaskukäskyn operandia korottaa yhdellä jokaisella laskukerralla samalla kun indeksirekisterin sisältö pienenee joka kerta yhdellä. Kun indeksirekisterissä on nolla, yhteenlasku on suoritettu kaikilla 50 luvulla ja ohjelmasilmukka päättyy. Tietokone on näin suorittanut 50 laskutoimitusta yhtä ainoaa yhteenlaskukäskyä käyttäen.

Kuvassa 118 esitetään indeksointisilmukan lohkokaavio. Ensimmäinen käsky sijoittaa suureen 50 indeksirekisteriin 4. Yhteenlaskukäsky, jonka operandina on 1051, sisältää operandiosassaan myös määrityksen, jonka mukaan annettua osoitetta muutetaan indeksirekisterissä 4 olevalla suureella. Seuraava käsky on haarautuminen indeksin mukaan, mikä tarkoittaa sitä, että indeksirekisterin sisällöstä vähennetään 1. Jos rekisterissä oleva luku on suurempi kuin nolla, yhteenlaskukäsky suoritetaan uudelleen. Jos rekisterissä on nolla, ohjelmaa jatketaan seuraavasta käskystä.

Indeksoinnin käyttö yksinkertaistaa huomattavasti toistuvien lasku- ym. toimintojen ohjelmointia ja vähentää tarvittavien käskyjen lukumäärää.

EPÄSUORAT OSOITTEET

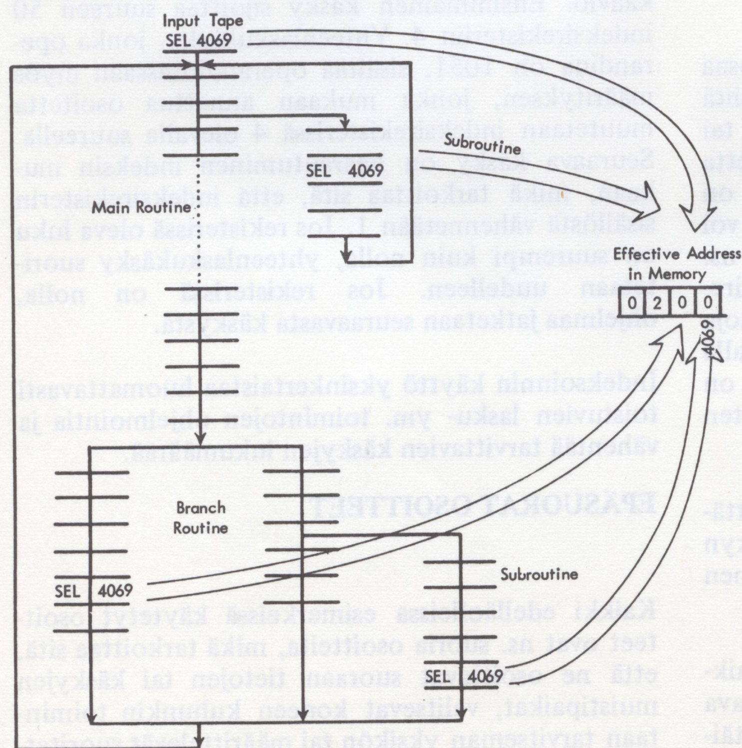
Kaikki edelläolleissa esimerkeissä käytetyt osoitteet ovat ns. suoria osoitteita, mikä tarkoittaa sitä, että ne osoittavat suoraan tietojen tai käskyjen muistipaikat, valitsevat koneen kuhunkin toimintaan tarvitseman yksikön tai määrittelevät suoritettavan valvontatoiminnan.



Kuva 118. Indeksisilmukka

Osoitteet voivat myös olla epäsuoria. Sellainen osoite viittaa vain muistipaikkaan, joka sisältää toisen osoitteen. Tämä toinen osoite puolestaan määrittää jonkin muistipaikan, koneyksikön tai ohjaustoiminnan.

Epäsuora osoite sopii erityisen hyvin osoitemodifioinnissa käytettäväksi. Ohjelmassa voi olla esimerkiksi välttämätöntä viitata käskyarvoihin, jotka muuttuvat toimeenpanovaiheen aikana. Ilman



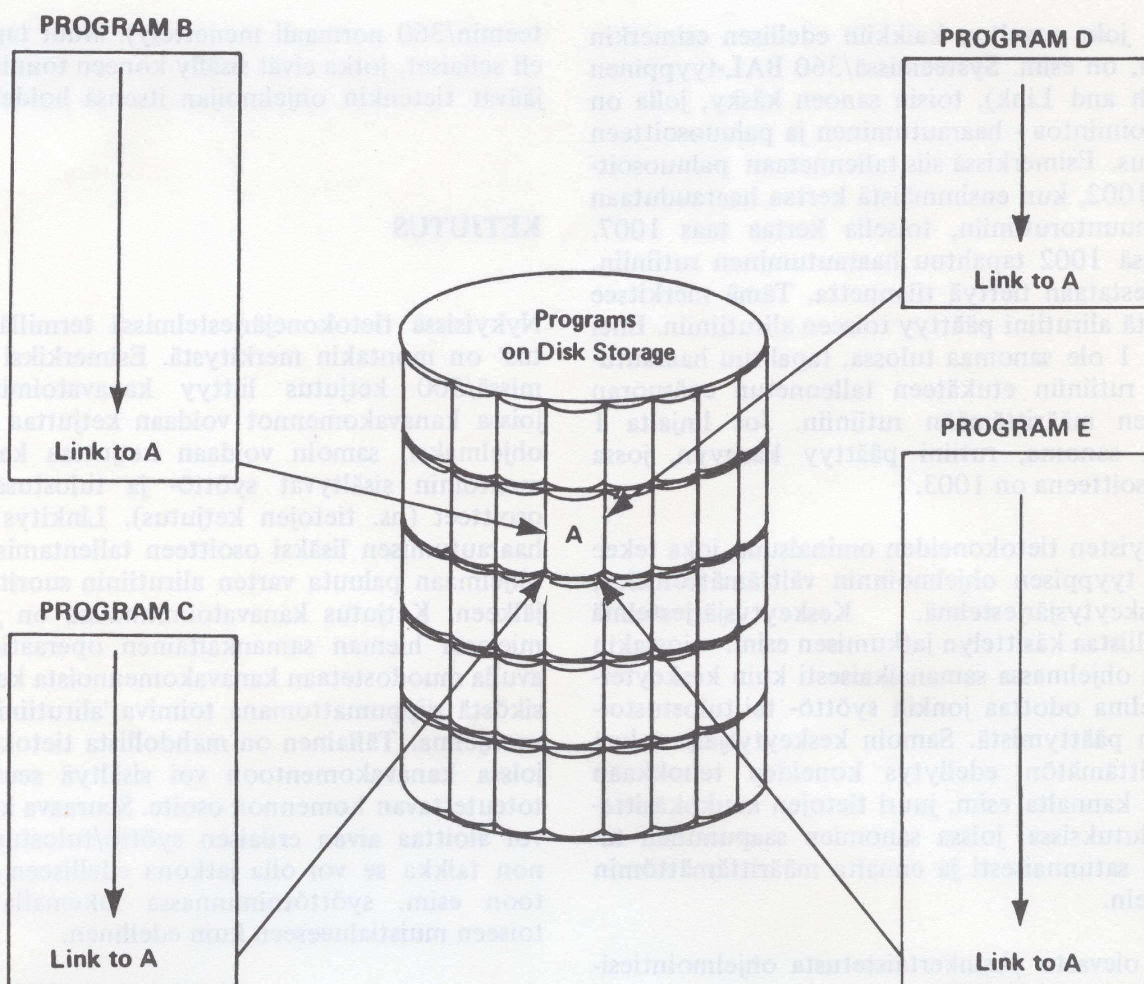
Kuva 119. Kaaviokuva epäsuorien osoitteiden käytöstä

epäsuoraa osoitteitusta tarvittaisiin joukko modifiointikäskyjä.

Kuitenkin jos käskyt osoittavat epäsuorasti yhteen muistipaikkaan, tämä muistipaikka voi sisältää yhden osoitteen, ohjelman käyttämien arvojen osoitteen. Sen tähden kaikkien käskyosoitteiden modifioimista varten on välttämätöntä modifioida vain sitä todellista osoitetta, johon käskyt viittaavat (kuva 119). Epäsuorat osoitteet kautta ohjelman voivat viitata yhteen todelliseen osoitteeseen. Kuvassa 119 jokainen epäsuorasti osoitettu käsky (SEL 4069) noutaisi muistipaikan 200 sisällön (eikä siis 4069:n sisältöä).

LINKITYS

Edellä on käsitelty ehdollista haarautumista ja viimeksi epäsuorien osoitteiden käyttöä. Nykyisissä tietokoneissa on yleensä mahdollista suorittaa yhdellä käskyllä haarautuminen alirutiiniin siten, että alirutiinin tultua toteutetuksi voidaan palata pääohjelmaan, haarautumiskäskyä seuraavaan käskyyn, tarvitsematta suorittaa osoitteiden tallennus- tai muita vastaavia toimintoja. Epäsuoraa osoitetekniikkaa käyttämällä voidaan kirjoittaa riippumattomia alirutiineja. 'Linkityskäsky' huolehtii



Kuva 120. Modulaarinen ohjelmarakenne ja linkitys

siitä, että joka kerran alirutiiniin tultaessa saadaan käyttöön myös oikea paluuosoite. Menetelmät, joilla eri tietokoneissa linkitys voidaan hoitaa, saattavat poiketa toisistaan. Seuraavassa eräs yksinkertaistettu periaatteellinen esimerkki.

Oletetaan, että kyseessä on tietojen kaukokäsittelyohjelma, jossa 1050-päätteellä tulostetaan monenlaisia raportteja, ja jossa syöttötietojen saamiseksi suoritetaan kiertokyselyä (polling) myös muiden päätteiden suhteen. Tehtävään kuuluu myös se, että kun mainittuja raportteja koskeva informaatio on käsitelty, se tallennetaan levyllä sijaitsevaan tiedostoon normaalissa BCD-muodossa. Joka kerta, kun sanoma on haettu ko. tiedostosta on ennen sen lähettämistä suoritettava muunto 1050-järjestelmän käyttämän BCD-koodin edellyttämään muotoon. Tehtävän suorittamiseksi tarvittava ohjelma näyttää esim. seuraavalta:

Käskyn numero		Vastaavan alirutiinin toiminta
1000	Haarautuminen, linkitys alirutiiniin (samaa käskyä käytetään kaikissa kohdissa)	Sanoman 1 luku levyltä
1001		1050-koodin muunto
1002		Linjalta 1 saapuneen syötteen käsittely (tarvittaessa)
1003		Sanoman 1 kirjoitus linjaan 2 kytketylle 1050-päätteelle
1004		Linjalta 3 saapuneen syötteen käsittely (tarvittaessa)
1005		Sanoman 2 luku levyltä
1006		1050-koodin muunto
1007		Linjalta 4 saapuneen syötteen käsittely (tarvittaessa)
1008		Sanoman 2 kirjoitus linjaan 2 kytketylle 1050-päätteelle

Käsky, joka soveltuu kaikkiin edellisen esimerkin kohtiin, on esim. Systeemissä/360 BAL-tyyppinen (Branch and Link), toisin sanoen käsky, jolla on kaksi toimintoa - haarautuminen ja paluuosoitteen tallennus. Esimerkissä siis tallennetaan paluuosoitteena 1002, kun ensimmäistä kertaa haaraudutaan 1050-muuntorutiiniin, toisella kertaa taas 1007. Käskyssä 1002 tapahtuu haarautuminen rutiiniin, jossa testataan tiettyä tilannetta. Tämä merkitsee sitä, että alirutiini päättyy toiseen alirutiiniin. Ellei linjalta 1 ole sanomaa tulossa, tapahtuu haarautuminen rutiiniin etukäteen tallennetun epäsuoran osoitteen määrittämään rutiiniin. Jos linjalta 1 luetaan sanoma, rutiini päättyy käskyyn, jossa paluuosoitteena on 1003.

Se nykyisten tietokoneiden ominaisuus, joka tekee tämän tyyppisen ohjelmoinnin välttämättömäksi, on keskeytysjärjestelmä. Keskeytysjärjestelmä mahdollistaa käsittelyn jatkumisen esim. jossakin muussa ohjelmassa samanaikaisesti kuin keskeytetty ohjelma odottaa jonkin syöttö- tai tulostustoiminnon päättymistä. Samoin keskeytysjärjestelmä on välttämätön edellytys koneiden tehokkaan käytön kannalta esim. juuri tietojen kaukokäsittelysovellutuksissa, joissa sanomien saapuminen tapahtuu satunnaisesti ja ennalta määrittämättömin aikavälein.

Edellä olevasta yksinkertaistetusta ohjelmointiesimerkistä voi havaita, että ohjelmien, joissa keskeytysjärjestelmää käytetään hyväksi, on koostuttava lyhyistä rutiineista. Tämä edellyttää myös hierarkista linkitysjärjestelmää ja osoitetaulukkoja siinä tapauksessa, ettei tietokoneen sisäisiin toimintoihin kuulu osoitteiden tallennus.

Linkitysjärjestelmän luonne jää melko paljon riippuvaksi tietokoneeen rakenteesta, ts. tietokoneen rakenteesta riippuu miten paljon jää ohjelmoijan itsensä huolehdittavaksi.

Keskeytykset, jotka tapahtuvat tietokoneen normaalitoimintojen tuloksina, (esim. kellolaitteen aiheuttamat keskeytykset, jotka tapahtuvat ennalta asetetuin aikavälein, taikka syöttö/tulostustoiminnon päättymisestä aiheutuvat keskeytykset) voivat aiheuttaa haarautumisen osoitteeseen, joka haetaan ennaltamäärätystä, kiinteästä muistipaikasta (Systeemin/360 normaali menettely). Muut tapaukset, eli sellaiset, jotka eivät sisälly koneen toimintoihin, jäävät tietenkin ohjelmoijan itsensä hoidettaviksi.

KETJUTUS

Nykyisissä tietokonejärjestelmissä termillä 'ketjutus' on montakin merkitystä. Esimerkiksi Systeemissä/360 ketjutus liittyy kanavatoimintoihin, joissa kanavakomennot voidaan ketjuttaa kanavaohjelmaksi, samoin voidaan ketjuttaa kanavakomentoihin sisältyvät syöttö- ja tulostusalueiden osoitteet (ns. tietojen ketjutus). Linkitys sisältää haarautumisen lisäksi osoitteen tallentamisen pääohjelmaan paluuta varten alirutiinin suorittamisen jälkeen. Ketjutus kanavatoiminnoissa on jossakin mielessä hieman samankaltainen operaatio - sen avulla muodostetaan kanavakomennoista keskusyksiköstä riippumattomana toimiva 'alirutiini', kanavaohjelma. Tällainen on mahdollista tietokoneissa, joissa kanavakomentoon voi sisältyä seuraavaksi toteutettavan komennon osoite. Seuraava komento voi aloittaa aivan erilaisen syöttö/tulostustoiminnon taikka se voi olla jatkona edelliseen komentoon esim. syöttötoiminnassa lukemalla tiedot toiseen muistialueeseen kuin edellinen.

Toinen ketjutuksen merkitys liittyy tietokoneisiin, jotka erityisesti on suunniteltu tietojen kaukokäsittelyä ja tietoliikennettä silmällä pitäen. Tässä tapauksessa ketjutus merkitsee automaattista muistitilan varausjärjestelmää, jonka puitteissa terminaaleista tulevat sanomat tai sanomalohkot tallennetaan mille tahansa vapaana olevalle alueelle siten, että järjestelmä automaattisesti sijoittaa kunkin muistialueen loppuun seuraavan ko. sanomalle varatun muistialueen osoitteen. Ketjutusjärjestelmän avulla vältytään varaamasta etukäteen jotakin tiettyä määrää muistialueita kutakin linjaa varten. Etukäteen lukkoon lyöty tilan varaus saattaa osoittautua riittämättömäksi taikka liian suureksi, jos linjalla onkin aktiviteettia odotettua vähemmän. Tässä mielessä ketjutus on dynaaminen ja tehokas tietojen puskurointimenetelmä (ks. myös jakso 'SYÖTTÖ- JA TULOSTUSLAITTEET', kappale 'Tietojen puskurointi').

Tietokoneiden mahdollisuudet laajenevat suunnatonta vauhtia - samoin laajenevat ihmisen mahdollisuudet käyttää tietokoneita hyväkseen yhä suuremmassa määrin kehittyneiden ohjausjärjestelmien ansiosta. Edistyminen tällä alueella on yhtä tärkeätä kuin itse koneiden kehitys. Tulevaisuuden tietokoneet eivät ole vain kehittyneempiä ja suurempia nopeudeltaan, muistikapasiteetiltaan jne. vaan hyvin suurelta osalta vaikuttavana tekijänä ovat näihin koneisiin kuuluvat kehittyneet ohjelmointi- ja käyttöjärjestelmät.

IBM:n ohjelmointikielet on kehitetty myös nimenomaan tulevaisuutta silmälläpitäen eikä vain nykyisiä laitteita ja sovellutuksia varten.

VALMISTELUTOIMINNOT

Ohjelma ei ole pelkästään järjestetty joukko yksityiskohtaisia käskyjä. Se on tulos ohjelmoijan ja tietokoneen välisestä yhteistyöstä.

Ohjelmoijan tehtävänä on annetun probleeman ratkaiseminen ja ratkaisun muokkaaminen tietokoneelle soveltuvaksi. Ensimmäiset työvaiheet ovat probleeman tarkka määritys, analysointi ja lohkokaavioiden laatiminen. Nämä tehtävät ovat yleensä riippumattomia tietokoneesta ja ohjelmointijärjestelmästä.

Seuraavassa on esitetty luettelonomaisesti eräitä seikkoja, jotka on otettava huomioon, olipa ohjelma kuinka yksinkertainen tahansa. On kuitenkin huomautettava, että hyvinkin monet alla olevista seikoista kuuluvat jo valmiina oleviin käyttöjärjestelmiin (eivätkä siten itse asiassa jää ohjelmoijan tehtäväksi). Käyttöjärjestelmiin palataan yksityiskohtaisemmin jäljempänä.

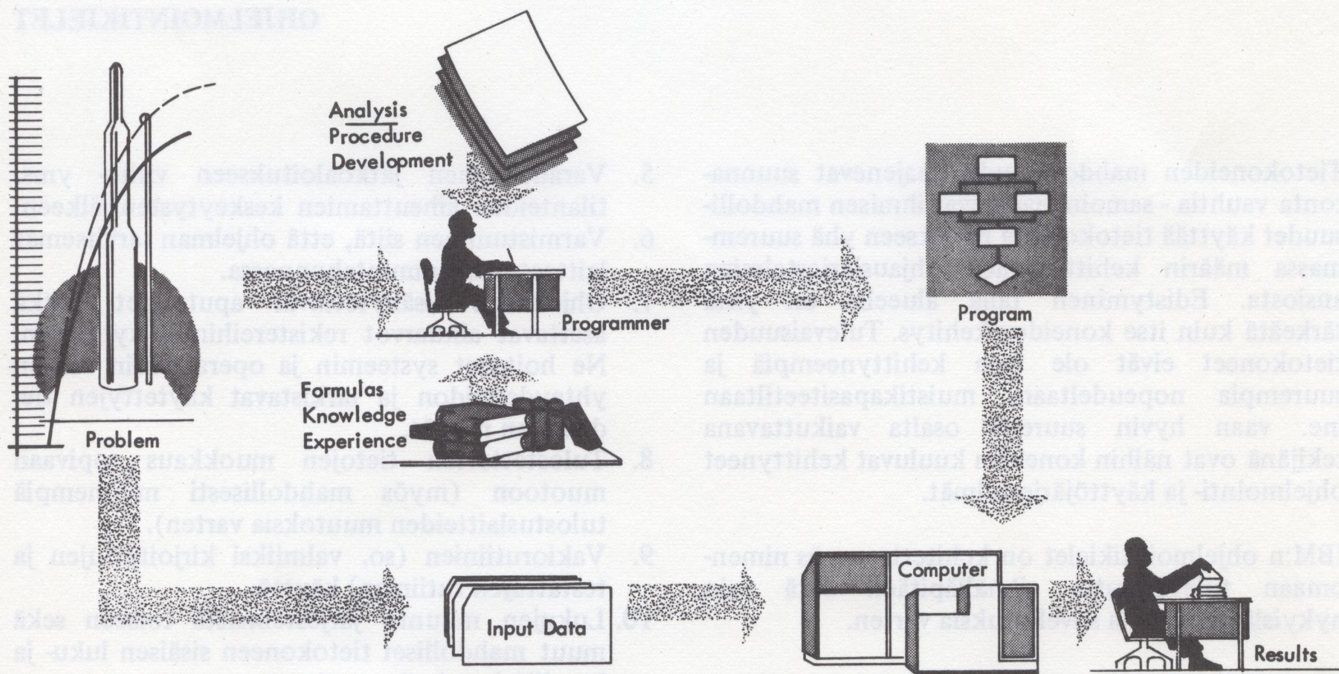
1. Muistin varaaminen syöttö- ja tulostustietoja, käskyjä ym. varten.
2. Käsiteltävän tietoaineksen muuntaminen soveltuvalle syöttövälineelle.
3. Huolehtiminen siitä, että kaikki ohjelman tarvitsemat taulukot, vakiot, tiedostot, ym. ovat käytettävissä.
4. Tarkistusmenetelmät ja yleensä luotettavuuteen liittyvät seikat.
5. Varautuminen jatkoaloitukseen virhe- yms. tilanteiden aiheuttamien keskeytysten jälkeen.
6. Varmistuminen siitä, että ohjelman tarvitsemat laitteet ovat toimintakunnossa.
7. Ohjelmaan sisällytettävät aputoimet, jotka asettavat alkuarvot rekistereihin ja kytkimiin. Ne hoitavat systeemin ja operaattorin välisen yhteydenpidon ja tarkistavat käytettyjen tiedostojen nimiöt.
8. Tulostettavien tietojen muokkaus sopivaan muotoon (myös mahdollisesti myöhempiä tulostuslaitteiden muutoksia varten).
9. Vakiorutiinien (so. valmiiksi kirjoitettujen ja testattujen rutiinien) käyttö.
10. Lukujen muunto järjestelmästä toiseen sekä muut mahdolliset tietokoneen sisäisen luku- ja koodijärjestelmän vaatimat muunnot.
11. Varsinainen tietojen käsittelyproseduuri, johon sisältyy myös poikkeuksellisten ja/tai virheellisten tietueiden käsittelyrutiinit.

KONEKIELINEN OHJELMOINTI

Kuvassa 121 on esitetty ohjelmointiprosessi kaaviona, josta käyvät myös selville ohjelmoijan ja tietokoneen väliset suhteet. Konekielisessä ohjelmoinnissa ongelman ratkaisu edellyttää sen hajoittamista mahdollisimman pieniin osiin, jotka vastaavat koneen toiminta-alkioita eli käskyjä. Ongelmasta itsestään muokataan tietokoneelle soveltuva syöttömateriaali, joka ohjelman avulla käsiteltynä antaa vaaditut tulokset.

Konekielisessä ohjelmoinnissa on kuitenkin omat hankaluutensa:

1. Kaikki käskyt on kirjoitettava konekielellä, mikä merkitsee esim. binääristä esitysmuotoa käyttävässä Systeemissä/360 ohjelmoinnin muodostuvan erityisen epäkäytännölliseksi.
2. Käskyt on kirjoitettava siinä järjestyksessä jossa tietokone tulee ne suorittamaan. Jos erehdyksessä jää käskyjä pois, kaikkia seuraavia käskyjä on muistissa muutettava tai sitten haarautumiskäskyin liitettävä puuttuvat käskyt ohjelmaan. Kaikki tähän käskyyn liittyvät työt, muistiosoitteiden ym. uudelleenlaskeminen jne. jää yksinomaan ohjelmoijan tehtäväksi.



Kuva 121. Probleeman muunto tietokoneohjelmaksi

3. Ohjelman logiikka pienimpiä yksityiskohtia myöten jää ohjelmoijan tehtäväksi.
4. Aikaisempien kokemusten - valmiiden ja testattujen rutiinien hyväksikäyttö on erittäin vaikeaa. Tällaisten rutiinien liittäminen uuteen ohjelmaan edellyttää oman 'linkitysrutiinin' rakentamista.
5. Ohjelmoijan on tunnettava tietokoneen toiminnot erittäin hyvin, koska konekielinen ohjelmointi edellyttää esim. erilaisten indikaattorien ja rekistereiden hyväksikäyttöä.

OHJELMOINTIJÄRJESTELMÄT

Monet edellä ilmitulleista vaikeuksista voidaan eliminoida osittain tai kokonaan kehittyneempien ohjelmointijärjestelmien avulla. Ohjelmointijärjestelmät merkitsevät sitä, että suoraan konekielelle suoritettava koodaus jää pois ja suuri osa ohjelmaan liittyvistä valmistelutoiminnoista voidaan jättää tietokoneen huoleksi.

Tietokone voidaan ohjelmoida siten, että varsinaiset käyttäjän ohjelmat voidaan esittää jollakin symbolikielellä, jonka kone muuntaa omalle sisäiselle kielelleen. Näin on kehitetty varsin edustava joukko ohjelmointikieliä, jotka käyttäjän kannalta ovat huomattavasti helpompia käyttää ja helpompia ymmärtää kuin konekieli.

SYMBOLIKIELI

Symbolisia ohjelmointikieliä käyttämällä ohjelmoija voi kirjoittaa käskyt hieman helpommin kuin varsinaisella konekielellä. Ohjelma pysyy tietyssä mielessä edelleen konekielisenä (sillä yksi symbolinen käsky vastaa yhtä konekielistä käskyä), mutta koneen sisäiset käskykoodit on korvattu muistamista helpottavin symbolein (mnemonics), esim. seuraavasti: kirjain A merkitsee yhteenlaskukäskyä, S vähennyslaskua, D jakolaskua, ST tallennusta jne. Samoin voidaan käskyjen operandit, viittaukset muistipaikkoihin, laitteisiin jne. esittää symbolisesti. Tietokone, joka toimii valmiiksi ohjelmoitujen käännösohjelmien valvonnassa, muuntaa symboliset käskyt vastaaviksi konekäskyiksi.

Vanhemmat symbolikielet perustuivat siihen että jokaisesta käskystä käännöksen tuloksena syntyi yksi konekielinen käsky. Esim. käsky
A REG1,184
jossa REG1 merkitsee yhtä Systeemin/360 kuudes-

tatoista yleisrekisteristä, on käännettynä konekielelle seuraavanlainen:

01011010000010000000000000010111000

Seuraavan kehitysvaiheen muodostivat makrokäskyt. Makrokäskyillä tarkoitetaan käskyä, joka edustaa suurempaa käskyjoukkoa, valmista rutiinia. Tämä merkitsi huomattavaa edistystä ohjelmointikielten käyttökelpoisuudessa. Tähän ajatukseen perustuvat nykyiset ohjelmointikielet, joita käyttämällä ohjelmoija voi antaa ohjeensa ja määräyksensä tietokoneelle englanninkielisin lausein, toisin sanoen kielellä, jota sekä ihminen että tietokone ymmärtävät.

Ohjelmointiin liittyvistä valmistelutehtävistä käännös konekielelle on ehkä tärkein, mutta ei ainoa. Vaaditun tuloksen saavuttamiseksi on tarvittavat käskyt toteutettava tietyssä järjestyksessä. Esim. yhteenlaskussa, jossa toisen tekijän on sijaittava rekisterissä, tarvitaan kaksi käskyä: latauskäsky, jolla yhteenlaskun toinen tekijä viedään rekisteriin sekä varsinainen yhteenlaskukäsky (ks. kuva 122).

Machine	Symbolic
01011000000100000000000011000000	L REG1, 384
01011010000100000000000011001000	A REG1, 392

Kuva 122. Kone- ja symbolikielisiä käskyjä

Kaikille lopullisille (konekielisille) käskyille on varattava tietty tila keskusmuistista. Edellisen yhteenlaskuesimerkin käskyt voisivat sijaita muistissa siten, että latauskäskyn osoitteena on 1000 mistä taas seuraa, että yhteenlaskukäskyn osoitteeksi tulee 1004, (koska latauskäskyn pituus on neljä tavua). Tällä tavoin jokainen käsky tulee 'nimetyksi' yksilöllisesti. Jos nyt joudutaan ohjelmaan lisäämään yksi tai useampia käskyjä, on lisäyskohdasta alkaen jokaisen käskyn osoitetta muutettava. Koska useimpiin ohjelmiin joudutaan ennemmin tai myöhemmin tekemään muutoksia, tulee osoitteiden määrittämisestä hankala ja työtä vaativa, mutta joka tapauksessa välttämätön osa ohjelmointia. Ongelmaa se ei nykyään enää muodosta, koska käännösohjelmat pystyvät hoitamaan myös tämän tehtävän. Ohjelmoija vain määrittää ohjelmansa alkuosoitteen (eli ensimmäisen käskyn osoitteen) ja käännösohjelma huolehtii siitä, että seuraavat käskyt saavat nousevat, peräkkäiset osoitteet.

Etu, joka symbolikielellä saavutetaan konekieleen verrattuna, on ilmeinen. Ja kuten aikaisemmin jo mainittiin, ei symboliikka rajoitu yksinomaan käskykoodeihin vaan soveltuu yhtä hyvin osoitteisiin, laiteviittauksiin jne. Samalla tavoin kuin käskytkin, kääntää käännösohjelma kaikki ohjelman sisältämät symboliset viittaukset todellisiksi osoitteiksi. Tuttu yhteenlaskuesimerkki voidaan nyt esittää symbolimuodossa (kuva 123). Käskyjen sisältöhän ei millään tavoin muutu: symbolisille operandeille on joka tapauksessa määritettävä todellinen, konekielinen osoite ja ero onkin vain siinä kuka määrittää osoitteet ja missä vaiheessa.

Instruction	
Operation Part	Address Part
L	REG1, A
A	REG1, B

Kuva 123. Symbolisia toimintakoodeja ja osoitteita

Jos käännösohjelmalle määritetään käännettävän ohjelman alkuosoitteeksi 1000, tulos olisi kuvan 124 mukainen. (Käskyt ja rekisterioperandit on säilytetty tässä symbolisessa muodossa ja osoitteet desimaalisina selvyyden vuoksi).

Instruction Location	Instruction	
	Operation Part	Address Part
1000	L	REG1, 4000
1004	A	REG1, 4004
.	.	.
.	.	.
4000	Value of A	
4004	Value of B	
.	.	.
.	.	.

Kuva 124. Esimerkki osoitemäärityksestä

Kuten jo on mainittu, edellä selostetun prosessin hoitamiseen tarvittavaa ohjelmaa nimitetään käännösohjelmaksi (translator, assembler). Ohjelmien kääntäminen tapahtuu yleensä siten, että ensin ladataan systeemin keskusmuistiin käännösohjelma (esim. systeemin kirjastosta). Tämän jälkeen

käännösohjelma lukee ohjelmoijan jollakin symbolikielellä kirjoittamat käskyt (eli käännettävän ohjelman). Käännöksen tuloksena syntynyt konekielinen ohjelma noudattaa luonnollisesti täysin samaa logiikkaa kuin lähtöohjelmakin. Käännetty ohjelma voidaan sijoittaa välittömästi muistiin tai se voidaan tulostaa esim. reikäkorteille.

OHJELMAN KÄÄNTÄMINEN

Ohjelmointikieli sisältää yleensä kaksi osaa:

1. Varsinaisen kielen ja siihen liittyvän kieliopin.
2. Käännösohjelman eli konekielisen ohjelman, jonka päätehtävänä on symbolisen ohjelman kääntäminen konekieliseen muotoon.

Käännettävää ohjelmaa eli käännösohjelman syötettä nimitetään lähtöohjelmaksi. Se on ohjelmoijan symbolisella ohjelmointijärjestelmän kielellä laatima kuvaus ongelmasta ja sen ratkaisusta. Lähtöohjelman kirjoittaminen edellyttää luonnollisesti hyvin tarkkaa ja täydellistä ongelman määrittämistä ja analysointia.

Tuloksena on 'lopullinen' ohjelma eli tulosohjelma, lähtöohjelman konekielinen käännös (aikaisemmin käytetty termi 'objektiohjelma' johtuu siitä, että konekieli on nimenomaan kullekin tietokoneelle ominainen sisäinen esitysmuoto ja käännös tapahtuu tietenkin sille kielelle, jota ohjelman ajossa käytettävä kone edellyttää).

Joissakin järjestelmissä käännösohjelman tulostama objektiohjelma (konekielinen ohjelma) on suoraan toteutettavassa muodossa. Toisissa taas tarvitaan erityinen linkitysohjelma (Linkage Editor), joka muokkaa tulosohjelman suoritettavaan muotoon. Linkitysajossa voidaan myös yhdistää erikseen käännettyjä ohjelmamoduuleja (alirutiineja, vakiotaulukoita tms.) yhdeksi ohjelmakokonaisuudeksi.

Kun ohjelma toteutetaan, sen tulosteet voidaan tallentaa magneettinauhalle tai -levystöille myöhemmin rivikirjoittimella suoritettavaa tulostusta varten. Tulostus voi tietenkin tapahtua myös suoraan rivikirjoittimella.

Testattua ohjelmaa käytetään jatkuvasti esim. kuukausittaisten raporttien laatimiseen (tuotantotilastot, varaston valvontaan liittyvät raportit jne.), jolloin tulokset ovat luonteeltaan samankaltaisia joka ajossa. Toisten ohjelmien tulokset saattavat joka ajossa olla erilaisia riippuen aivan siitä, mihin tarkoitukseen ohjelmaa käytetään (yleisluontoiset optimointiohjelmat tms.) - onko tarkoituksena laskea lentokoneen siiven aerodynaamisesti edullisin muoto, höyrypannun putkien edullisin sijoittelu tai muuta vastaavaa.

KONEENLÄHEISET OHJELMOINTIKIELET

Koneenläheisissä ohjelmointikielissä tietokoneen toiminnot kuvataan symbolisin koodein. Samoin esitetään symbolein käskyjen operandit (tietokenttien nimet, osoitteet jne.) Käännösohjelma suorittaa sitten symbolisten koodien ja nimen muunnon konekäskyiksi ja todellisiksi osoitteiksi.

Systeemin/360 Assembler-kieli on eräs tällaisista koneenläheisistä ohjelmointikielistä. Käskyt kirjoitetaan erityisille lomakkeille (ks. kuva 125). Lomakkeen jokaisesta rivistä lävistetään reikäkortti. Lomakkeen pystysarakkeet vastaavat reikäkortin sarakkeita.

Lomakkeella on tilaa myös ohjelman tunnustiedoille sekä lävistysohjeille. Näitä tietoja ei kuitenkaan lävistetä korteille. Varsinaista ohjelmaa varten tarkoitettu osa lomaketta jakautuu edelleen kahteen osaan: käskyosaan (sarakkeet 1-71) ja tunnusosaan (73-80). Käskyosassa on tilaa käskyn neljälle komponentille (joiden kaikkien ei välttämättä tarvitse sisältyä jokaiseen käskyyn). Alkaen vasemmalta voidaan lomakkeen riville kirjoittaa käskyn nimi (8 merkkiä), toimintakoodi (5 merkkiä), operandi(t) ja/tai käskyyn liittyvä huomautus, kommentti (56 merkkiä).

Symbolisen nimen avulla ohjelmoija voi identifioida haluamansa käskyn. Nimien käyttö on täysin ohjelmoijan harkinnasta ja tarpeista riippuvaa eikä siis pakollista. Jos käsky nimetään, nimen on alettava sarakkeesta 1, sen pituus on yhdestä kahdeksaan merkkiä eikä se saa sisältää tyhjää.

Toimintakoodi on muistamista helpottava symboli, joka edustaa käskyä tai jotain käännösohjelmalle annettavaa ohjetta. Toimintakoodi on kaikkiin käskyihin ja lauseisiin kuuluva pakollinen osa. Ellei käskyllä ole nimeä, voidaan toimintakoodi kirjoittaa lomakkeelle mihin kohtaan tahansa alkaen sarakkeesta 2, jos taas käskyllä on nimi, on nimen ja toimintakoodin välissä oltava ainakin yksi tyhjä sarake. (Samat säännöt pätevät käskyjen muihinkin komponentteihin - merkintätapa on siten melko

IBM — IBM System/360 Assembler Coding Form

PROGRAM PROGA
PROGRAMMER J.J. JONES

```
PROGA    START 256
BEGIN    BALR  11,0
         USING *,11
         L     2,DATA            LOAD REGISTER 2
         A     2,CON             ADD 10
         SLA   2,1               THIS HAS EFFECT OF MULTIPLYING BY 2
         S     2,DATA+4          NOTE RELATIVE ADDRESSING
         ST    2,RESULT
         L     6,BIN1
         A     6,BIN2
         CVD   6,DEC             CONVERT TO DECIMAL
         EOJ                     END OF JOB
DATA     DC    F'25'
         DC    F'15'
CON      DC    F'10'
RESULT   DS    F
BIN1     DC    F'12'
BIN2     DC    F'78'
DEC      DS    D
         END   BEGIN
```

Kuva 125. Esimerkki Assembler-kielisestä ohjelmasta

vapaamuotoinen - eli komponenttien on oltava yllä esitetyssä järjestyksessä ja niitä erottamassa on oltava vähintään yksi tyhjä sarake, mutta muuten niiden sijaintia lomakkeen rivillä ei ole rajoitettu).

Toisaalta ohjelman luettavuus ja ymmärrettävyys paranee sillä, että käytetään jotakin standardia, esimerkiksi merkitsemällä käskykoodi aina sarakkeelle 10, aloittamalla operandit sarakkeelta 106 jne.

Operandit edustavat tietoja, joihin käskyjen toiminta kohdistuu. Operandit sisältävät myös informaatiota, jolla kuvataan tietokentän pituus, tiedon muoto jne. Käskyssä voi olla yksi tai useampia operandeja. Operandit erotetaan toisistaan pilkulla eikä operandilistaan saa sisältyä tyhjiä sarakkeita. Kuten mainittu, operandit edustavat tietoja, joilla käskyt operoivat, toisin sanoen operandit voivat olla rekistereitä, osoitteita tai ohjelmavakioita. Käännösohjelman toimintakäskyjen operandit sisältävät niinikään tietoja, joita käännösohjelma käyttää hyväkseen halutun toiminnon suorittamiseksi.

Huomautukset, kommentit ovat lisäinformaatiota, joka voi liittyä tiettyyn käskyyn tai kokonaiseen rutiiniin, mutta niitä ei käännöksessä mitenkään huomioida. Niiden avulla ohjelman luettavuus paranee ja ohjelmalistasta saadaan dokumentti, jolla on käyttöarvoa myöhemmin mahdollisesti tehtäviä muutoksia silmälläpitäen. Sen lisäksi, että käskyriville voidaan kirjoittaa lyhyt kommentti, voidaan kokonainen rivi varata kommentteja varten sijoittamalla tähti sarakkeeseen 1. Kommenttikorttien määrää lähtöohjelmassa ei ole mitenkään rajoitettu.

Edellinen ei tietenkään voinut olla täydellinen esitys Systeemin/360 Assembler-kielestä, mutta näiden perusseikkojen pohjalta ohjelman kirjoittamisen ja koodauksen periaatteita lienee ainakin hieman helpompi ymmärtää.

MAKROKÄSKYT

Ohjelmointijärjestelmän tehokkuutta voidaan huomattavasti lisätä laajentamalla jonkin verran käännösohjelman toimintoja.

Systeemiin kuuluvan makrokielen avulla ohjelmoija

voi mukavasti ja melko helposti itsekin laatia makrokäskyjä - käskyjä, joissa toimintakoodi (ja siihen liittyvät operandit eli parametrit) itse asiassa edustaa käskyjoukkoa, rutiinia jonkin tehtävän suorittamiseksi. Nykyisiin järjestelmiin kuuluu olennaisena osana suuri joukko IBM:n valmiiksi laatimia makrokäskyjä, mutta kuten sanottu, mikään ei estä ohjelmoijaa laatimasta myös omia makrokäskyjä omien vaatimustensa mukaisia toimintoja varten.

Makrokäskyn määritys tapahtuu vain kerran ja sen edustama käskyjoukko voidaan 'kutsua' ohjelmaan kirjoittamalla makrokäskyn toimintakoodi (makron 'nimi') ohjelmaan samalla tavoin kuin muutkin käskyt. Tällä voidaan helpottaa ohjelmointia, vähentää koodausvirheitä sekä varmistua siitä, että tietty toiminto, (joka siis on määritetty makrokäskyksi) aina toteutetaan samalla tavoin.

Makrokäskyn toimintakoodi

Makrotoimintakoodi (tai makrokäsky) on siis lähtöohjelman käsky tai lause, joka edustaa joukkoa konekielisiä käskyjä. Käännösohjelma käsittelee makrokäskyjä periaatteessa samalla tavoin kuin muitakin käskyjä.

Joka kerran kun käännösohjelma kohtaa lähtöohjelmassa makrokäskyn se liittää lähtöohjelmaan kyseisen makromäärityksen sisältämät käskyt ja käsittelee sen jälkeen näitä 'generoituja' käskyjä niin kuin mitä tahansa Assembler-kielistä käskyä.

Makromääritys

Ennen kuin makrokäsky voidaan kääntää, on käännösohjelman käytettävissä oltava itse makro (tai makromääritys). Makro koostuu lauseista tai käskyistä, joilla ilmoitetaan käännösohjelmalle makrokäskyn toimintakoodi, sen muoto sekä ne käskyt, jotka käännösohjelman tulee makrokäskyn kohdatessaan liittää lähtöohjelmaan (siis ennen varsinaista käännösprosessia).

Makrokirjasto

Makromäärityksestä saadaan 'pysyvä' ja se voidaan asettaa kaikkien muidenkin ohjelmien käytettäväksi sijoittamalla se erityiseen makrokirjastoon. Makrokirjasto, joka sijaitsee jollakin 'on-line' -muistilaitteella (esim. levyllä) on käännösohjelman käytettävissä oleva installaation kaikkien makrojen muodostama kokoelma.

Makrokäskyn muotoilu

Makrokäskyn perusteella generoitujen käskyjen määrän ja sisällön ei tarvitse välttämättä olla vakio, vaan niitä voi tapauksesta riippuen vaihdella erityisten ehdollisten käännösohjelmaa ohjaavien käskyjen avulla. Näillä voidaan makron sisältämien käskyjen muotoa, lukumäärää tms. muuttaa tarpeen mukaan.

Jokaiseen ns. tehtävän läheiseen kieleen kuuluu myös oma menetelmänsä, jolla makroja kirjoitetaan. Jotkut kielet ovat kirjoitustavaltaan hyvinkin lähellä normaalia englanninkieltä. Näihin tehtävän läheisiin kieliin kuuluvat COBOL, FORTRAN sekä uusimpana jäsenenä yleiskieli PL/I. Termi 'tehtävän läheinen' jo selittää näiden kielten luonteen - ne ovat paljon lähempänä tietokoneiden avulla suoritettavia tehtäviä kuin minkään nimenomaisen koneen ominaisuuksia ja toimintoja.

COBOL

COBOLin (Common Business-Oriented Language) käyttö ohjelmointikielenä ei millään tavoin muuta sitä, että tulostusohjelman edelleen on oltava konekielinen. Kääntäjän tehtävät pysyvät edelleen periaatteessa samoina, mutta kieli, jolla ohjelmoija kirjoittaa lähtöohjelman, ei enää juuri lainkaan muistuta konekieltä. Samoin jää ohjelmoijan kannalta sivuseikaksi se, miten COBOL-ohjelman kääntäminen itse asiassa suoritetaan.

Yksinkertainen esimerkki valaissee parhaiten tehtävän läheisen ohjelmointijärjestelmän perusperiaatteita. Oletetaan, että on laskettava esim. verotusta varten verotettava tulo. Tekijöinä tässä esimerkissä voisivat olla kaksi muuttujaa: TULOT joihin nyt pitäisi lisätä OSINGOT. COBOL-kielellä yhteenlasku voidaan esittää seuraavalla lauseella:

ADD OSINGOT TO TULOT.

Ennen kuin COBOL-kääntäjä voi tulkita tämän lauseen, on sille kuitenkin annettava hieman lisäinformaatiota. Ohjelmoijan on esimerkiksi ilmoitettava nimet TULOT ja OSINGOT tietyssä ohjelman osassa, joka nimenomaan on varattu tietojen määritystä varten (Data Division). Määri-

tyksiin sisältyvät myös muuttujien pituus, numerollinen esitysmuoto jne.

Kun kääntäjä kohtaa tällaisen lauseen, sillä on käytössään myös erinäisiä lisätietoja sekä kääntäjässä itsessään 'sisäänrakennettuina' että käyttöjärjestelmän kirjastoissa, jotka ovat avuksi käännöksen suorittamisessa. (On kuitenkin huomattava, että varsinaisen käännösprosessin kulku on täysin käytetystä tietokoneesta riippuvainen eikä ohjelmoijan missään tapauksessa tarvitse kääntäjän yksityiskohtia tunteakaan).

Ensimmäiseksi kääntäjä tutkii sanan ADD. Kääntäjään kuuluu oma sanastonsa, joka sisältää ns. COBOL-sanat eli sellaiset, joilla on oma tarkoin määritetty merkityksensä tässä kielessä. Jos kääntäjä havaitsee ADDin olevan eräs näistä sanoista, se pystyy myös tulkitsemaan sanan siten, että tulostusohjelmaan on tätä sanaa vastaavaan kohtaan sijoitettava yhteenlaskun suorittamiseksi tarvittavat konekieliset käskyt.

Seuraavaksi kääntäjä tutkii sanan OSINGOT. Koska kääntäjällä on jo tästä sanasta informaatiota käytettävissään (määritysten vuoksi), nimittäin tiedot siitä mistä tämän muuttujan löytää ja minkä tyyppistä tietoa se sisältää, se sijoittaa muuttujan todellisen (konekielisen) osoitteen tulosohjelman tarvittaviin käskyihin.

Kun kääntäjä löytää sanan TO, se jälleen palaa tutkimaan omaa sanastoaan. Tässä tapauksessa TO johtaa suoraan sanaan TULOT, ja määrittää siten muuttujan nimeltä TULOT yhteenlaskun tulosalueeksi.

Esimerkkilauseen viimeisen sanan, TULOT, joka edustaa erästä muuttujaa, käsittely on samanlainen kuin toisenkin muuttujan (OSINGOT).

Eräs osanen on vielä käsittelemättä, nimittäin piste, joka päättää COBOL-lauseen. Pisteen vaikutus on COBOLissa siten sama kuin esim. suomen kielessä. Kääntäjälle piste ilmoittaa, että se on käsitellyt ne muuttujat joita ADD-verbi koskee.

Edellä selostettiin niitä toimenpiteitä, joita kääntäjä suorittaa kehittäessään tulostusohjelmaa. Tämä toiminta ei tosin ehkä aina ole täsmälleen samanlaista, koska koneet poikkeavat toisistaan ja kääntäjät puolestaan on erikseen suunniteltu kullekin konetyypille. Lopullinen tulos on tietenkin aina sama: konekieliset käskyt, jotka laskevat yhteen muuttujat OSINGOT ja TULOT.

FORTRAN

FORTRAN (Formula Translation) ohjelmointijärjestelmänä on periaatteessa COBOLin kaltainen. Perusero on kielellisessä ilmaisussa, jota lähtöohjelmissa käytetään. COBOLissa käytettävä englannin kieli on FORTRANissa korvattu matemaattisilla ilmaisuilla. Edellinen COBOL-esimerkkilause

ADD OSINGOT TO TULOT.

kirjoitettaisiin FORTRAN-kielellä seuraavasti:

TULOT = TULOT + OSINGOT.

Joissakin FORTRAN-kääntäjissä saattaa olla rajoituksia esim. muuttujien nimien pituuksiin nähden, jolloin esimerkilause jouduttaisiin lyhentämään muotoon:

TUL = TUL + OS

Mutta käännöksen lopputulokseenhan tällainen ei vaikuta. Lausehan on kääntäjän kannalta määräys kehittää lauseen sisältämien toimintojen toteuttamiseksi tarvittavat konekieliset käskyt. On huomattava, että esimerkissä ei ainoastaan suoriteta yhteenlaskua vaan määritetään myös paikka, johon tulos tallennetaan. Jos muuttujan TULOT alkuperäinen arvo oli esim. 10000 ja muuttujan OSINGOT arvo 15, korvautuu TULOT laskutoimituksen jälkeen uudella arvolla 10015.

Lause voidaan kirjoittaa myös siten, ettei kummankaan muuttujan arvo laskutoimituksessa muutu, esimerkiksi:

VTULOT = TULOT + OSINGOT

Lauseen muuttaminen tähän muotoon aiheuttaa sen, että yhteenlaskun tuloksen tallentamista varten kehittää käännösohjelma uuden muuttujan (VTULOT), jolloin yhteenlasku ei myöskään muuta muiden muuttujien arvoja.

PL/I

Kuten jo aikaisemmassa yhteydessä on mainittu, Systeemi/360 on tietojenkäsittelyjärjestelmänä eräänlainen yleiskone - se soveltuu yhtä hyvin niin kaupalliseen kuin tieteellistekniseenkin tietojenkäsittelyyn. Edellä käsitellyt ohjelmointikielet, 'tieteellinen' FORTRAN ja 'kaupallinen' COBOL eivät

kumpikaan sellaisinaan pysty käyttämään Systeemin/360 kaikkia mahdollisuuksia.

Myös kehitys, joka on johtanut tietokoneiden käyttöön yhä laajenevassa määrin päätöksen teon apuvälineenä, ennusteiden laatimisessa jne. sekä tietojen kaukokäsittelyssä, vaatii myös ns. kaupalliselta ohjelmoinnilta entistä enemmän 'tieteellistä näkemystä' ja entistä monipuolisempia proseduureja ongelman eteen. Itse asiassa traditionaalinen ero kaupallisten ja tieteellisten ohjelmien välillä on kokonaan häviämässä. Tämä juuri antoi aiheen luoda sellainen yleiskieli, jota voitaisiin käyttää sekä kaupallisiin että tieteellisiin sovellutuksiin. Kieli on nimeltään PL/I. Kuvien 126 ja 127 esimerkki antaa yleiskuvan kielestä ja sen eräistä ominaisuuksista.

PL/I:ssä kuten FORTRAN-kielessäkin, aritmeettiset toiminnot voidaan kuvata operaattorien(+,-, /, ★, ja ★★) avulla. Esimerkkilauseessa TULOT = TULOT + OSINGOT PL/I eroaa FORTRAN ista vain siinä, että PL/I -lause päättyy puolipisteeseen, (;) sekä siinä, että kirjoitussäännöissä ei ole juuri lainkaan rajoituksia.

Yhteenlasku voidaan esittää myös hieman COBOL-kielen tapaan:

SUMMA = ADD (OSINGOT,TULOT,8,2);

Yhteenlaskun tulos sijoitetaan kenttään nimeltä SUMMA, jonka pituus on 8 positiota mukaan luettuna 2 desimaalia.

ADD, DIVIDE, MULTIPLY, ATAN, COMPLEX, ERF, INDEX, MAX, MIN, ROUND, SUM ja TAN ovat eräitä valmiita PL/I-funktioita, joiden avulla voidaan käsitellä yksityisiä tietoelementtejä, vektoreita, matriiseja tai struktuureja. PL/I-kielen avainsanat eivät ole varattuja sanoja siinä mielessä kuin COBOL issa. Ohjelman kirjoitussäännöt ovat hyvin väljät. Esimerkiksi kuvassa 127 sama ohjelma (josta PL/I:ssä käytetään termiä PROCEDURE) on esitetty riveillä 1-8 ja 9-12.

Sellaisten sanojen kuin PAGE, SKIP, LINE, COLUMN, PAGE SIZE, LINE SIZE avulla voidaan tulostettavan listan muoto määrittää yksityiskohtaisesti.

RPG

RPG (Report Program Generator) eli raporttiohjelman kehittäjä on myös eräs tehtävän läheinen ohjelmointimenetelmä ja lisäksi kaikkein yksinkertaisin käyttäjän kannalta. Tiedostojen määrityksiin ja yleensä syöttö/tulostustoiminnan hoitamiseen, jotka muissa menetelmissä saattavat vaatia ohjelmoijalta melko paljonkin vaivaa, tarvitaan RPG:ssa vain muutaman rivin täyttäminen erityiselle määrityslomakkeelle.

Syötemäärityslomakkeella kuvataan seuraavia tietoja:

1. Syöttötiedosto(t)
2. Kunkin tiedoston sisältämät erilaiset tietuetyypit
3. Tietueiden sisältämät tietokentät.

ja tulostemäärityslomakkeella vastaavasti:

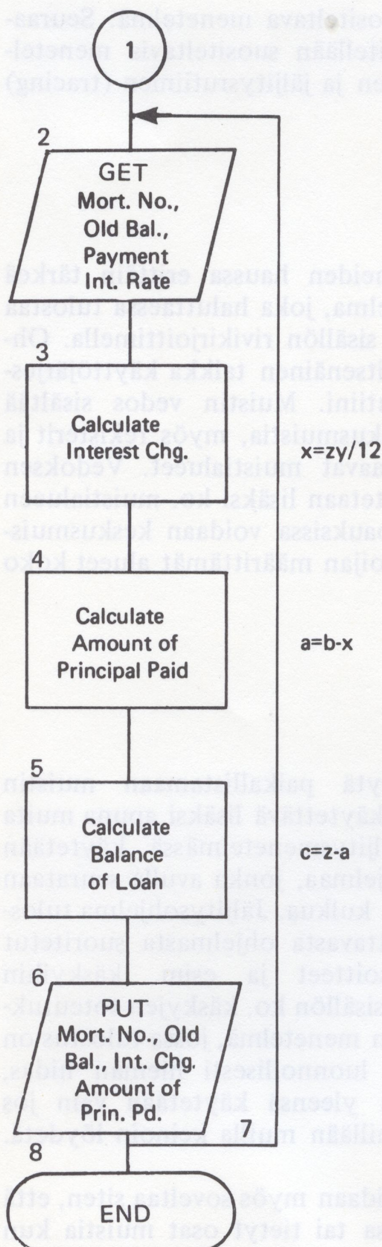
1. Tulostustiedostot (so. kirjoitettavat raportit, summakortit jne.)
2. Tulostettavat tietokentät
3. Raportteihin mahdollisesti liitettävät otsikkorivit
4. Tietokenttien ja raporttien muotoiluun liittyvät seikat.

Tiedostojen määrityslomakkeella ilmoitetaan lisätietoja ohjelman tarvitsemista tiedostoista (esim. laite, missä tiedosto sijaitsee).

Toimintojen määrityslomakkeella kuvataan laskutoimitukset, vertailut jne. eli yleisesti sanoen kaikki muut paitsi syöttö- ja tulostustoiminnot, jotka siis ovat eräänlaisia automaattisia toimintoja RPG:ssa. Tällä lomakkeella esitetään ohjelman logiikka määrittämällä

1. milloin jokin toiminto suoritetaan (esim. summakeräilyt per ohjaustaso),
2. mikä toiminto suoritetaan,
3. mitä testejä tuloksien suhteen suoritetaan.

Riippuen Systeemin/360 käyttöjärjestelmästä on RPG:ssa käytettävissä 22-27 erilaista toimintakoodia, mm. yhteen-, vähennys-, kerto- ja jakolaskukäskyt, vertailu, siirto, haarautumiskäsky, etumerkin testauskäskyt, aliohjelmien liittämiseen tarvittavat käskyt jne.



Kuva 126. Osa lainojen käsittelyohjelman lohkokaaviota

```
BILLING: PROCEDURE OPTIONS (MAIN);
    NEXTCARD:GET DATA (MORNO, OBAL, PAYM, RATE);
                CHARGE= OBAL*RATE/12;
                PRINPAID= PAYM-CHARGE;
                BALANCE = OBAL-PRINPAID;
            PUT DATA (MORNO,OBAL, CHARGE, PRINPAID, BALANCE;
    GO TO NEXTCARD;
END BILLING
```

```
BILLING:PROCEDURE OPTIONS (MAIN);NEXTCARD:GET DATA (MORNO, OBAL, PAYM,
RATE);CHARGE=OBAL*RATE/12;PRINPAID=PAYM-CHARGE;BALANCE=
OBAL-PRINPAID;PUT DATA (MORNO, OBAL, CHARGE, PRINPAID,
BALANCE) ;GO TO NEXTCARD;END BILLING;
```

```
BOB: PROCEDURE OPTIONS (MAIN);  /*THIS IS A COMMENT*/
    D:GET DATA (N, Z, B, Y);  /*COMMENTS MAY BE PUT*/
        X=Z*Y/12;     /*WHEREVER BLANKS ARE ALLOWED*/
        A=B-X;
        C=Z-A;
      PUT DATA (N, Z, X, A, C);
    GO TO D;
END BOB;

BOB: PROCEDURE; D:GET DATA (N, Z, B, Y); X=Z*Y/12; A=B-X;
C=Z-A; PUT DATA (N, Z, X, A, C); GO TO D; END BOB;
```

Kuva 127. Kuvan 126 rutiini PL/I-kielellä koodattuna

OHJELMAN TESTAUS

Kun ohjelma on saatu käännetyksi, on seuraavana vaiheena testaus eli 'koeajo' testiaineistolla. Tällä varmistetaan se, että ohjelma ennen varsinaista käyttöönottoaan on kunnossa, ts. ei sisällä loogisia virheitä ja että tulokset tosiaan vastaavat odotuksia. Testin tulostusmahdollisuuksia on kaksi. Ensimmäinen - ja toivottavasti ainoa - mahdollisuus on se, että ohjelma toimii odotusten mukaisesti ja voidaan ottaa normaalikäyttöön. Toinen mahdollisuus on se, että 'testi onnistui, mutta ohjelma ei toiminut'. Syitä tähän voi olla monia. Tavallisin on se, että lähtöohjelmassa on virheitä ja puutteellisuuksia, joita käännösohjelman tarkistusrutiinit eivät pysty havaitsemaan.

Tällaisia ohjelmoijan erehdyksiä on vaikeampi

välttää kuin voisi olla odotettavissa. Itse asiassa hyvin harvat ohjelmat toimivat asianmukaisesti ensimmäisellä testikerralla. Jotkut ohjelmat saattavat vaatia hyvinkin monia testiajoja ennen kuin kaikki virheet on löydetty ja korjattu. Karkeimmat virheet tosin paljastuvat jo käännöksen aikana (kääntäjän monipuolisten tarkistusrutiinien ansiosta). Esimerkkejä tällaisista virheistä ovat viittaaminen sellaiseen symboliseen osoitteeseen, jota ei ole lainkaan määritetty, eksponenttiaritmetiikan käskyjen käyttö tavallisilla desimaaliluvuilla jne.

Tietokoneen itsensä tekemät virheet ovat hyvin harvinaisia ja ne paljastuvat yleensä hyvin helposti koneen sisäänrakennetun virhetarkistusjärjestelmän ja erilaisten indikaattorien ansiosta. Monin verroin vaikeampaa on ohjelmoijan tekemien virheiden havaitseminen ja varsinkin virheen todellisen syyn löytäminen.

Testauksen tekniikkaa

Kuten jo aikaisemmin selostettiin, ohjelmien laatimiseen voidaan käyttää suoraan konekieltä, symbolista konekieltä tai jotakin tehtävän läheistä kieltä (PL/I, COBOL, FORTRAN, RPG). On selvää, että suoraan konekielelle kirjoitetut ohjelmat ovat alttiimpia virheille (ja lisäksi ohjelmointi yleensä vaikeampaa) kuin jotakin muuta kieltä käytettäessä. Voidaan sanoa, että lähtöohjelman virheiden määrä on tietyssä mielessä kääntäen verrannollinen käytettyyn ohjelmointimenetelmään, so. mitä 'korkeampaa' ohjelmointikieltä käytetään, sitä helpompaa ohjelmointi on (samoin kuin lähtöohjelman virheiden selvittäminen). Toisaalta taas testauksen yhteydessä esiintyvien virheiden selvittäminen saattaa symbolisella konekielellä kirjoitetuissa ohjelmissa olla helpompaa kuin ns. lausekielisissä (COBOL, FORTRAN tms.) ohjelmissa (koska symbolinen käsky vastaa yhtä tietokoneen toiminta-alkiota, konekielistä käskyä, kun taas esim. COBOL-lause lähinnä vastaa makrokäskyä). Moniin lausekieliin sisältyvät testaus- ja virheiden jäljitysrutiinit ovat ohjelmoijalle suureksi avuksi.

Testaus- ja virheenhakumenetelmiä on useita, joilla kullakin on omat etunsa ja haittapuolensa. Se, mitä menetelmää kulloinkin käytetään, riippuu suuressa määrin ohjelmoijan arvioinneista miten laajasta virheestä on kysymys ja missä kohden ohjelmaa virhe todennäköisesti on. Testaaminen 'nappulatekniikalla' eli tietokoneen ohjauspöydän avulla on yleensä pelkkää koneajan tuhlausta eikä se tämän takia ole mikään suositeltava menetelmä. Seuraavassa sen sijaan käsitellään suositeltavia menetelmiä, muistin vedoksen ja jäljitysrutiinien (tracing) hyväksikäyttöä.

Muistin listaus

Testauksessa ja virheiden haussa erittäin tärkeä työväline on apuohjelma, joka haluttaessa tulostaa koko keskusmuistin sisällön rivikirjoittimella. Ohjelma voi olla joko itsenäinen taikka käyttöjärjestelmään kuuluva rutiini. Muistin vedos sisältää paitsi varsinaista keskusmuistia, myös rekisterit ja systeemin tilaa kuvaavat muistialueet. Vedoksen kullekin riville kirjoitetaan lisäksi ko. muistialueen osoite. Joissakin tapauksissa voidaan keskusmuistista tulostaa ohjelmoijan määrittämät alueet koko muistin sijasta.

Jäljitys
(Tracing)

Ellei virheitä pystytä paikallistamaan muistin vedoksen avulla, on käytettävä lisäksi apuna muita menetelmiä. Ns. jäljitysmenetelmässä käytetään erityistä rutiinia, ohjelmaa, jonka avulla seurataan testattavan ohjelman kulkua. Jäljitysohjelma tulostaa tavallisesti testattavasta ohjelmasta suoritetut käskyt, näiden osoitteet ja esim. käskyihin liittyvien rekisterien sisällön ko. käskyjen toteutuksen jälkeen. Tällainen menetelmä, jossa tulostus on käskykohtainen, on luonnollisesti hieman hidas, mutta toisaalta sitä yleensä käytetään vain jos ohjelmavirhettä ei millään muilla keinoin löydetä.

Jäljitystekniikkaa voidaan myös soveltaa siten, että tulostetaan tietty osa tai tietyt osat muistia kun testattavassa ohjelmassa kohdataan jokin ennalta määrätty käsky. Tällä tavoin saadaan ohjelmasta eräänlainen tilannekuva. Esim. haarautumiskäskyt voidaan määrittää jäljityksen kohteiksi, jolloin tuloksena saadaan luettelo toteutetuista haarautumiskäskyistä sekä joukko tilannekuvia keskusmuistista samassa järjestyksessä. Näin pystytään seuraamaan käskyjen todellista toteutusjärjestystä ja sen vaikutusta eri muistialueisiin.

Yhteenveto

Se miten hyvin ohjelman tarkistamisessa ja testauksessa onnistutaan, riippuu hyvin monista seikoista. Tiettyjen perussääntöjen noudattaminen

helpottaa tämän joskus hyvinkin vaivalloisen, mutta kuitenkin välttämättömän työvaiheen suorittamista.

1. Ohjelma on syytä dokumentoida niin hyvin, että muutkin kuin ohjelmoija itse pystyvät saamaan selville mitä ohjelma oikeastaan tekee ja miten se toimii.
2. Ennen käännöstä ja testausta on lähtöohjelma ehdottomasti tarkistettava vertailemalla sitä olemassa olevaan dokumentointiin (esim. lohkokaavioihin).
3. On syytä jo etukäteen ohjelmassa varautua muutoksiin ja erilaisiin testausrutiineihin jättämällä tilaa näitä varten.
4. On tärkeää tietää mitä testausmenetelmiä on käytettävissä ja miten niitä voidaan käyttää hyväksi. Ei yleensä kannata jättäytyä pelkästään yhden menetelmän varaan ja hylätä kaikkia muita.
5. Ohjelman testaus päättyy ja ohjelman voidaan sanoa olevan kunnossa vasta silloin, kun se toimii niin kuin sen edellytetäänkin toimivan kaikissa olosuhteissa ja myös kaikki poikkeustilanteet huomioidaan.
6. On myös muistettava, ettei onnistunut testi aina takaa todellisen ajon onnistumista. Testimateriaali saattaa olla epätäydellistä tai muissa suhteissa poiketa todellisesta aineistosta. Tästä syystä testiaineiston valmistaminen on erittäin tärkeä ja huolellisuutta vaativa työ.

SYÖTTÖ- JA TULOSTUSTOIMINTOJEN OHJAUSJÄRJESTELMÄT

Makrokäskyjen säästäessä paljon vaivaa muodostaa toisaalta syöttö- ja tulostustoimintojen hoitaminen melkoisen ongelman ja runsaasti työtä kokeneellekin ohjelmoijalle. Yksinkertaista olisi tietenkin käsitellä ohjelmassa tietue kerrallaan - lukea tietue, käsitellä se ja kirjoittaa tulokset. Magneettinauhojen ja -levyjen tehokas hyväksikäyttö edellyttää kuitenkin tietueiden ryhmittelyä sopiviin lohkoihin sekä käsittelyn ja syöttö/tulostustoimintojen limittämistä keskenään.

Näiden ongelmien ratkaisemiseksi kehitettiin syöttö/tulostustoimintojen ohjausjärjestelmä (IOCS). Ohjelmointijärjestelmää ja syöttö/tulostustoimintojen ohjausjärjestelmää hyväksikäyttämällä ohjelmasta tulee ohjelmoijan kannalta joukko peräkkäisiä toimintoja ja ohjelmoija voi keskittyä päätehtäväänsä, käsittelyongelman ratkaisemiseen. Ohjelmoijan tehtäväksi jää vain tiedostojen kuvaus - ohjausjärjestelmän rutiinit huolehtivat lopusta.

Tiedostojen määrityslauseet, jotka yleensä ovat makroja, kuuluvat oleellisena osana ohjelmointijärjestelmään. Ja kuten jo mainittiin, ohjausjärjestelmän avulla ohjelmoija vapautuu lähes täysin ongelmista, jotka liittyvät fyysiseen syöttö/tulostustoimintaan, laitteisiin jne. ja voi keskittyä varsinaisiin käsittelyongelmiin.

Levymuistien tultua markkinoille kävi syöttö- ja tulostustoimintojen ohjelmointi yhä monimutkaisemmaksi. Näiden laitteiden tärkein ominaisuus, mahdollisuus poimintakäsittelyyn, korosti entisestään toimintojen asianmukaisen järjestelyn tärkeyttä (ja kuten sanottu, myös monimutkaisuutta).

Järjestelyltään peräkkäisissä magneettinauhatiedostoissa voidaan seuraava lohko lukea muistiin samaan aikaan kuin edellistä käsitellään ja siten ennen kuin sitä varsinaisesti tarvitaan. Poimintamuistilaitteilla tämä ei aina käy päinsä, koska 'seuraava' tietue ei ehkä fyysisesti olekaan seuraava vaan saattaa sijaita missä tahansa. Tähän on löydettävissä useita ratkaisuja, sekä yksinkertaisia että monimutkaisia. Eräs ratkaisu on ohjelman jakaminen useihin aliohjelmiin siten, että jokainen aliohjelma pystyy käsittelemään tietyn tyyppisen tietueen taikka paikallistamaan seuraavan tietyn tyyppisen tietueen. Ohjausjärjestelmästä haarautuminen näihin aliohjelmiin voi tapahtua enemmän tai vähemmän satunnaisessa järjestyksessä riippuen siitä, mikä monista haettavista tietueista löydetään ensimmäiseksi. Tämä on itse asiassa eräänlainen moniajojärjestelmä, jonka puitteissa useat eri ohjelmat suorittavat tehtävänsä osittain järjestelyrutiinin määrittämässä järjestyksessä ja sen valvonnassa.

Useimpiin syöttö/tulostustoimintojen ohjausjärjestelmiin kuuluvia ominaisuuksia käsitellään seuraavissa kappaleissa. Mikään järjestelmä ei ehkä sisällä kaikkia mainittuja ominaisuuksia, mutta kaikkiin kuuluu ainakin osa niistä. Joukossa on sellaisia ominaisuuksia, jotka eivät ole yleisesti käytössä lähinnä sen vuoksi, että ne osoittavat miten laajasta toiminta-alueesta on kysymys.

Nykyään nämä ohjausjärjestelmät ovat huomattavasti muuttuneet ja kehittyneet siitä, mitä ne alunperin olivat. Niistä on tullut olennainen osa koko tietojenkäsittelysysteemiä valvovasta käyttö-

järjestelmästä (erinomainen esimerkki tästä on Systeemin/360 käyttöjärjestelmä OS/360), jonka puitteissa ne hoitavat syöttö- ja tulostustoiminnassa tietojen sekä sisäisen että laitteisiin liittyvän käsittelyn.

Syötön ja tulostuksen järjestely (Scheduling)

Jotkut tietokoneet hoitavat syötön ja tulostuksen sarjamaisesti. Mitään käsittelyä ei voida suorittaa ennen kuin syöttö- tai tulostustoiminto on päättynyt ja päinvastoin, syöttö- ja tulostustoimintoja ei suoriteta sillä aikaa kun keskusyksikkö käsittelee tietoja. Toisissa tietokoneissa taas voidaan syöttö/tulostustoimintoja ja käsittelytoimintoja suorittaa samanaikaisesti antamalla keskusyksikön jatkaa käsittelyä välittömästi sen jälkeen kun syöttö/tulostustoiminto on aloitettu. Tietokone on ikään kuin jaettu kahteen (tai useampaan) itsenäisesti toimivaan komponenttiin. Tällaisella samanaikaisella (asynkronisella) syöttö/tulostustoiminnalla, joka koskee kaikkia tietokoneeseen liitettyjä laitteita, voidaan eliminoida keskusyksikön tarpeettomat odotusajat (joita sarjamaisessa toiminnassa syntyy aina kun tietojen siirtoa tapahtuu muistista tai muistiin).

Syöttö/tulostustoimintojen ohjausjärjestelmän, IOCS:n avulla pystytään käyttämään hyväksi monimutkaisiakin laitteita, jotka tekevät nykyaikaisesta tietojenkäsittelyjärjestelmästä todella tehokkaan.

Tietueiden jaksotus

Magneettinauhat ja -levystöt menettävät suuren kapasiteettinsa tarjoamat edut, jos tiedot niille tallennetaan lyhyinä lohkoina. Jos esim. lohkon pituus on 80 tavua, ei enää kannata puhua tiedostosta sanan varsinaisessa merkityksessä vaan paremminkin lohkovälikokoelmasta (kolme neljännestä koko tiedoston käytössä olevasta magneettinauhan alueesta sisältää näitä lohkovälejä). Jaksottamalla yhteen lohkoon aina esim. 10 tietuetta jää entisistä kymmenestä lohkovälistä jäljelle vain yksi ja tulos on se, että alkuperäiselle alueelle tallennettavien tietojen määrä moninkertaistuu edelliseen verrattuna. Koska nauha kulkee vakionopeudella lukupään kautta, nauhayksikkö käyttää nyt ajastaan enemmän todellisten tietojen lukuun ja vähemmän lohkovälien ohittamiseen. Lopputulos tästä kaikesta on todellisen tietojensiirtonopeuden huomattava kasvu.

Menetelmän teho pienenee ellei tietueiden lukua tapahdu niin usein, että nauhayksikkö sen johdosta pysyy jatkuvasti toiminnassa. Tässä tapauksessa rajoittavan tekijän muodostaa keskusyksikön nopeus ja ohjelma on siten käsittelysidonnainen. Jos taas asianlaita on päinvastainen, ohjelma on siirräntäsidonnainen ja nyt voidaan lohkotusta säätelemällä pienentää yhden loogisen tietueen lukemiseen tarvittavaa keskimääräistä aikaa.

Koska syöttö- ja tulostuslaitteet yleensä edellyttävät, että kerralla luetaan tai kirjoitetaan kokonainen lohko (esim. nauhaa ei voida pysäyttää keskelle lohkoa), on tulostettavat tietueet jaksotettava eli koottava lohkoihin. Tietojen syötössä tarvitaan päinvastainen toimenpide, lohkon purkaminen loogisiin tietueihin, joita annetaan yksi kerrallaan käyttöohjelman käsiteltäväksi.

Standardivirherutiinit

Monet syöttö/tulostustoiminnassa esiintyvät tilanteet ovat poikkeuksia normaalista tietueen luvusta tai kirjoituksesta. Ohjelmoijaa ei yleensä kiinnosta kaikkien tällaisten mahdollisuuksien huomioon ottaminen jokaisen syöttö/tulostuskäskyn kohdalla. Olisi epämielekästä esim. joka kerta nauhatiedostoon viitattaessa testata onko tiedoston loppu jo saavutettu. Tällainen johtaisi siihen, että vain kerran esiin tuleva tilanne, tiedoston loppu, vaatisi ehkä yhtä paljon ohjelmointia kuin itse tietueen käsittely. Monet epätavalliset tai poikkeukselliset tilanteet ovat luonteeltaan yleisiä ja vaativat aina samankaltaisen käsittelyn, joten niiden hoitaminen syöttö/tulostustoimintojen ohjausjärjestelmän yleisten virherutiinien avulla on ohjelmoijan kannalta erittäin käytännöllinen ratkaisu.

Seuraavissa kappaleissa on käsitelty joitakin ohjausjärjestelmän avulla hoidettavista virhe- ja poikkeustilanteista.

Virheiden korjausmenettely

Jos tietojen siirto laitteen ja keskusyksikön välillä epäonnistuu sitä ensimmäistä kertaa yritettäessä, tietyillä menetelmillä voidaan auttaa tilanteen selvittämistä siten, että ohjelma voisi keskeytyksettä jatkaa toimintaansa. Eräs vakiomenettely saattai-

si esimerkiksi magneettinauhoilla olla se, että virheellisesti nauhalle kirjoitettu tietue pyyhitään ensin kokonaan pois ja kirjoitetaan koko tietue uudelleen. Jos kirjoitus nyt tapahtui virheettömästi, ei muita toimenpiteitä tietenkään tarvita. Vain jos uudelleenkirjoitus jatkuvasti epäonnistuu, on tieto tilanteesta toimitettava operaattorille ja/tai ohjelmalle. Tähän tapaan useimmat virhetilanteet voidaan korjata automaattisesti ilman ohjelmalle asetettavia lisävaatimuksia.

Nauhan ja tiedoston loppu

Kun kaikki magneettinauhan tietueet on käsitelty, syntyy nauhan (nauhakelan) lopputilanne. Jos lisäksi on tiedoston (joka voi käsittää useitakin nauhakeloja) kaikki tietueet käsitelty, syntyy tiedoston lopputilanne, josta on annettava ilmoitus käyttöohjelmalle.

Väärämittainen tietue

Jos luettava tietue koneen toimintahäiriöstä tai ohjelmointivirheestä johtuen on virheellisen mittainen, tilanne havaitaan jo ohjausjärjestelmässä ja ryhdytään asian vaatimiin toimenpiteisiin. Jos virhe on sellainen, ettei esillä olevaa työtä voida jatkaa, voidaan automaattisesti siirtyä seuraavaan työhön taikka systeemi voidaan myös pysäyttää sen jälkeen kun se on kirjoittanut asianmukaisen virheilmoituksen. Joissakin tapauksissa riittää tilanteesta ilmoittaminen käyttöohjelmalle, jossa sitten lopullisesti ratkaistaan miten tilannetta käsitellään.

Magneettinauhanimiöt

Suuren, tuhansien tai kymmenientuhansien markkojen arvoisia tietoja sisältävän nauhakirjaston ylläpito edellyttää huolellisuutta ja vastuuntunnetta tämän tehtävän suorittajalta, jotta voitaisiin taata tietojen säilyminen vahingoittumattomina. Huolimaton operaattori, joka epähuomiossa sijoittaa nauhayksikköön jonkin sovellutuksen perustiedoston sisältävän nauhakelan (ja asettaa kelalle kirjoituksen mahdollistavan renkaan ja poistaa näin erään suojausmahdollisuuden), jolle kirjoitetaan jotakin tietoa, voi menettelyllään aiheuttaa koko sovellutukselle tavattoman uuria vahinkoja (nauhalle kirjoittaminenhan tuhoaa siellä ennestään olevat tiedot).

Lähinnä tällaisten tapausten varalle on kehitetty nimiöjärjestelmä. Kullekin kelalle tallennetaan ensimmäiseksi tietolohkoksi alkunimiö, jonka sisältämä informaatio identifioi ko. nauhan käyttöohjelman tai ohjausjärjestelmän nimiötarkistusrutiinia varten. Vertaamalla tarvittavan nauhan tunnusta tai nimeä yksikössä olevalle nauhalle tallennettuun tunnukseen voidaan varmistua siitä, että käsitellään juuri oikeata nauhaa ja ettei nauhoja turmella vahingossa.

Levystönimiöt

Levystön nimiöintiä varten on olemassa erityiset apuohjelmat, jotka tämän lisäksi suorittavat myös muut levystön alustuksen vaatimat tehtävät. Nimiöiden avulla kukin taltio ja tiedosto on identifioitavissa.

APUOHJELMAT

Apuohjelmat on laadittu tiettyjen kaikille sovellutuksille yhteisten ja yleisten toimintojen suorittamiseksi.

Tällaisia apuohjelmia ovat tiedostojen siirtoon laitteelta toiselle käytettävät ohjelmat (korteilta nauhalle, levylle tms.) sekä tietueiden vertailuun käytettävät ohjelmat. Tämän tyyppisistä sekä IBM:n että asiakkaiden laatimista ohjelmista koostuva kirjasto palvelee tietokoneinstallaatioita samaan tapaan kuin julkiset kirjastot yleisöä.

KIRJASTOT

Kirjastojen järjestely ja tietojen saanti kirjastoista ovat tietokoneinstallaation joustavan toiminnan kannalta ensiarvoisen tärkeitä seikkoja. Koska magneettinauhojen asetus ja nauhakirjastojen hoitaminen vaatii työtä, aikaa, huolellisuutta ja lisäksi fyysistä säilytystilaa, pyritään nykyisin yhä enemmän säilyttämään kirjastoja levystöillä, suurilla poimintamuistilaitteilla, rummuilla ja kennomuisteilla.

Systeemissä/360 kullakin syöttö/tulostuslaitteella on kiinteä osoite, jota ei voida muuttaa. Edelleen kunkin laitetyypin puitteissa jokaisella erillisellä alayksiköllä - levystö, tietokenno, magneettinauha,

rumpu jne. - on oma kiinteä osoitteensa. Näistä alayksiköistä tai tietojen tallennusvälineistä käytetään nimitystä volyymi eli taltio.

Käyttöjärjestelmän tehtävänä on jakaa kaikki systeemiin saapuvat ohjelmat ja muu tietoaines taltioille, pitää kirjaa taltioista ja kunkin taltion sisällöstä sekä taltioissa olevasta käyttämättömästä tilasta.





KÄYTTÖJÄRJESTELMÄT

Käyttöjärjestelmä on työohjelmiston ja ohjausohjelmiston muodostama integroitu kokonaisuus, jonka tarkoituksena on tietokoneen toimintatehon parantaminen. Käyttäjän valmistamien ohjauskorttien tai -lauseiden ohjaamana käyttöjärjestelmä yhden työn tultua valmiiksi siirtyy automaattisesti seuraavaan työhön, jolloin töiden asetusajat ja operaattorin asioihin puuttuminen jäävät mahdollisimman vähiin. Koko systeemin ja operaattorin välinen yhteydenpito tapahtuu käyttöjärjestelmän kautta eli siis toisin kuin vanhemmissa tietokonejärjestelmissä.

Nykyisissä nopeissa ja tehokkaissa tietokoneissa muodostavat töiden asetusajat suhteellisen suuren osan työn kokonaisajasta. Tämä johtaa siihen, että suuri ja varsin kallis tietokone odottaa joutilaana sen ajan, jonka uuden työn käsittelyvalmiiksi saattaminen kestää. Samoin vielä ohjelman toteutuksen aikana hyvin monet systeemin komponentit saattavat jäädä toimettomiksi. Kaikista systeemin käytettävissä olevista resursseista vain osaa saatetaan tarvita tietyn työn suorittamiseen. Käyttöjärjestelmän avulla voidaan työt niputtaa (jolloin niiden suoritus tapahtuu yhtäjaksoisena toimintana) asetusaikojen minimoimiseksi. Käyttöjärjestelmä hallitsee koko tietokonejärjestelmää. Se pystyy 'kutsumaan' keskusmuistiin kaikki tarvittavat ohjelmat, aliohjelmat, tiedot jne. Jotkut käyttöjärjestelmät pystyvät lisäksi järjestelemään kaikki suoritettavat työt (valitsemalla sopivimman suoritusjärjestyksen) ja jakamalla käytettävissä olevat resurssit erittäin tehokkaasti esim. siten, että kaksi tai useampia töitä suoritetaan samanaikaisesti (so. kaksi tai useampia toisistaan riippumattomia ohjelmia). Tällainen järjestely kulkee nimellä moniajo (multiprogramming). Näitä toteutettavia ohjelmia voivat olla käyttöjärjestelmän omat komponentit, esim. käännösohjelmat (FORTRAN, COBOL, PL/I, RPG) tai käyttöohjelmat. Lyhyesti sanottuna käyttöjärjestelmän avulla voidaan tietokonetta käyttää hyväksi mahdollisimman tehokkaasti.

Systeemissä/360 on neljä erilaista käyttöjärjestelmää. Mikä näistä sitten parhaiten soveltuu kullekin installaatiolle, riippuu käytettävissä olevan tietokoneen koosta, muusta laitteistosta, sekä siitä mitä lisäominaisuuksia ja -funktioita käyttöjärjestelmältä vaaditaan. Jonkin käyttöjärjestelmän sisältämä funktio tai erikoispiirre ei välttämättä sisälly muihin järjestelmiin.

Mainitut neljä käyttöjärjestelmää ovat BOS/360 (Basic Operating System), TOS/360 (Tape Operating System), DOS/360 (Disk Operating System) ja OS/360 (Operating System). Joitakin niiden rakenteellisista yhtäläisyyksistä ja eroavaisuuksista käsitellään seuraavissa kappaleissa.

Tarkoituksena on esittää lyhyt katsaus niihin seikkoihin, jotka ensisijaisesti vaikuttavat käyttöjärjestelmän valintaan tietylle laitteistolle. Käyttöjärjestelmien kaikkien ominaisuuksien sisällyttäminen tämän tyyppiseen kirjaseen ei ole tarkoituksenmukaista. Se, että jonkin käyttöjärjestelmän kohdalla on jokin ominaisuus jätetty mainitsematta, ei välttämättä merkitse sitä että ko. ominaisuus ei tosiaan sisälly käyttöjärjestelmään.

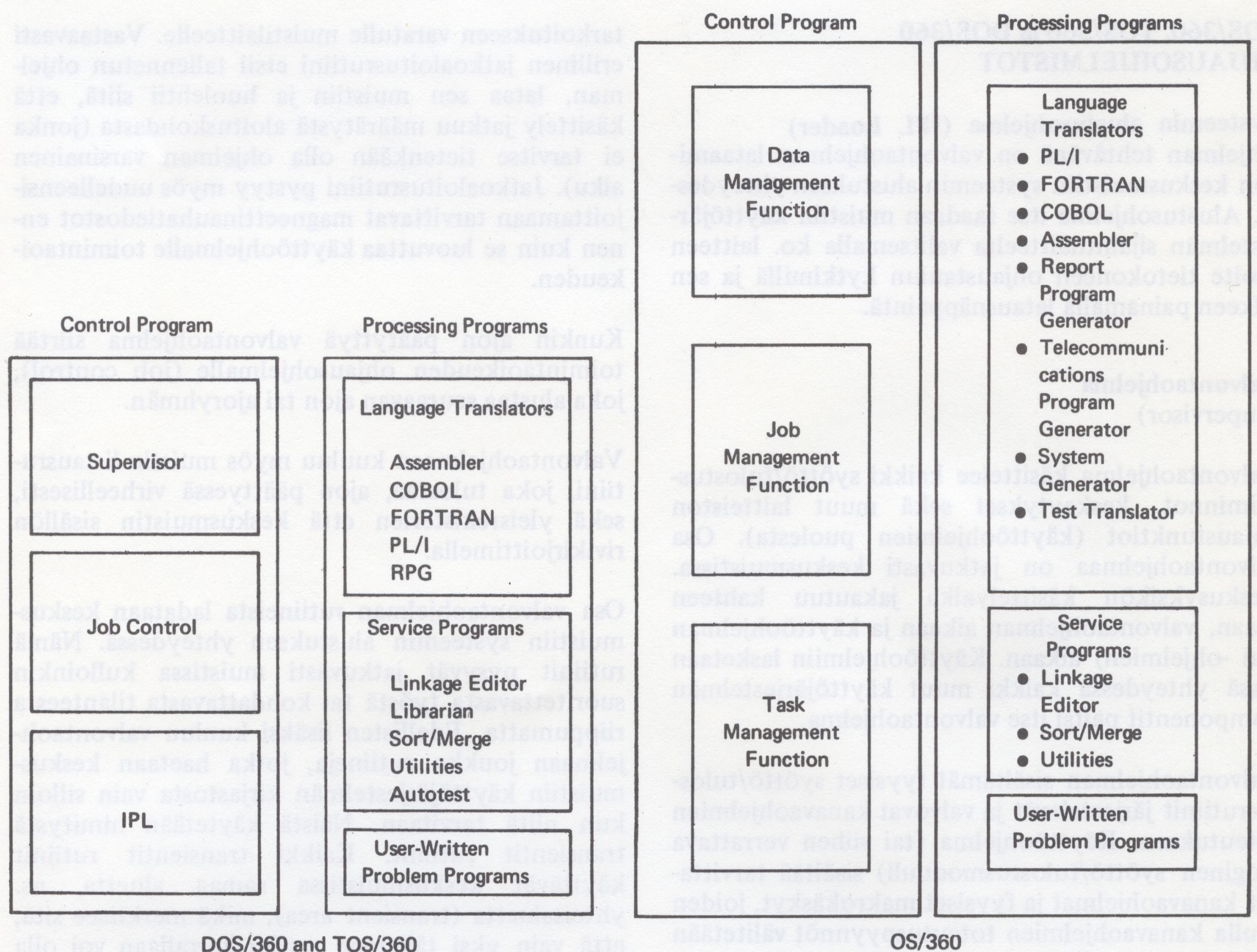
Kaikki käyttöjärjestelmät voidaan jakaa kahteen peruskomponenttiin, jotka ovat ohjausohjelmisto ja työohjelmisto.

Ohjausohjelmien tehtävänä on valvoa työohjelmien toteutusta, huolehtia tietojen paikantamisesta, tallentamisesta ja esille saannista sekä töiden järjestelystä siten, että niiden suorituksen välinen hukka-aika jää mahdollisimman pieneksi.

Työohjelmistoon kuuluvat käännösohjelmat, apuohjelmat ja installaation omat käyttöohjelmat, joita kaikkia ohjelmoija voi käyttää hyväkseen suunnitellessaan laitteistolla suoritettavaa työtä sekä nimenomaan pyrkiessään helpottamaan ohjelmiin liittyviä valmistelutöitä.

Käyttäjä voi myös liittää systeemiin omia käännösohjelmiaan ja apuohjelmiaan, joiden käyttö on samanlaista kuin käyttöjärjestelmän 'omien' komponenttienkin. Kuvassa 128 on esitetty käyttöjärjestelmien DOS, TOS ja OS perusfunktiot ja -komponentit.

Kaikissa neljässä käyttöjärjestelmässä suoritettavalla työllä ymmärretään ohjausjärjestelmän valvonnassa tapahtuvaa työnipun tai työjonon sisältämien tehtävien suorittamista. Työ eli ajoryhmä (job)



Kuva 128. Käyttöjärjestelmien komponentit

saattaa sisältää yhden tai useampia ajoja (job step). Ajoryhmää voidaan pitää jonona, jonka muodostavat loogisesti toisiinsa liittyvät, määrätyssä järjestyksessä toteutettavat ohjelmat. Ajoryhmän puitteissa ajot (ohjelmat) suoritetaan peräkkäin eikä samanaikaisesti.

Kunkin työn suorittamiseen tarvittava informaatio annetaan käyttöjärjestelmälle ohjauskorteilla. Ohjausohjelma lukee nämä kortit, jotka mm. ilmoittavat mikä tai mitkä ohjelmat halutaan toteuttaa ja missä järjestyksessä sekä määrittävät ajokohtaiset syöttö- ja tulostuslaitteet. Ohjauskorteilla ilmoitetaan nippuajoon (stacked job) tarvittavat tiedot.

BOS/360, TOS/360 ja DOS/360 OHJAUSOHJELMISTOT

Systeemin alustusohjelma (IPL Loader)
Ohjelman tehtävänä on valvontaohjelman lataaminen keskusmuistiin systeemin alustuksen yhteydessä. Alustusohjelma itse saadaan muistiin käyttöjärjestelmän sijaintilaitteelta valitsemalla ko. laitteen osoite tietokoneen ohjaustaulun kytkimillä ja sen jälkeen painamalla latausnäppäintä.

Valvontaohjelma
(Supervisor)

Valvontaohjelma käsittelee kaikki syöttö/tulostustoiminnot, keskeytykset sekä muut laitteiston ohjausfunktiot (käyttöohjelmien puolesta). Osa valvontaohjelmaa on jatkuvasti keskusmuistissa. Keskusyksikön käsittelyaika jakautuu kahteen osaan, valvontaohjelman aikaan ja käyttöohjelman (tai -ohjelmien) aikaan. Käyttöohjelmiin lasketaan tässä yhteydessä kaikki muut käyttöjärjestelmän komponentit paitsi itse valvontaohjelma.

Valvontaohjelman sisältämät fyysiset syöttö/tulostusrutiinit järjestelevät ja valvovat kanavaohjelmien toteutuksen. Käyttöohjelma (tai siihen verrattava looginen syöttö/tulostusmoduuli) sisältää tarvittavat kanavaohjelmat ja fyysiset makrokäskyt, joiden avulla kanavaohjelmien toteutuspyynnöt välitetään valvontaohjelmalle.

Valvontaohjelma aloittaa syöttö/tulostustoiminnon ja palauttaa toimintaoikeuden takaisin käyttöohjelmalle. Kun syöttö/tulostustoiminto on päättynyt, valvontaohjelma toteaa ja käsittelee mahdolliset virheet ja toimintahäiriöt. Tällä funktiolla on suuri merkitys sikäli, ettei käyttöohjelmaan tarvitse yleensä liittää minkäänlaisia virherutiineja.

Valvontaohjelma sisältää myös jatkoaloitusrutiinit, joiden avulla käyttöohjelman toteutus voidaan keskeyttää halutussa kohdassa, tallentaa ohjelman sen hetkinen tila ja jatkaa sen toteuttamista keskeytyskohdasta joskus myöhemmin. Ohjelman toiminnan ei välttämättä tarvitse keskeytyä tarkistuspisteen (check-point) kohdalla. Tarkistuspisteitä voidaan ottaa niin paljon kuin halutaan ja jatkaa jälleen käsittelyä tarkistuspisteestä eteenpäin. Tarkistuspisteen tallennusta varten on olemassa makrokäsky nimeltä CHKPT, jonka suorittaminen kutsuu keskusmuistiin tarkistuspisterutiinin ja tämä puolestaan kirjoittaa koko käyttöohjelman sekä muut jatkoaloitukseen tarvittavat tiedot tarkoitukseen varatulle muistilaitteelle. Vastaavasti erillinen jatkoaloitusrutiini etsii tallennetun ohjelman, lataa sen muistiin ja huolehtii siitä, että käsittely jatkuu määrätystä aloituskohdasta (jonka ei tarvitse tietenkään olla ohjelman varsinainen alku). Jatkoaloitusrutiini pystyy myös uudelleensijoittamaan tarvittavat magneettinauhatiedostot ennen kuin se luovuttaa käyttöohjelmalle toimintaoikeuden.

Kunkin ajon päätyttyä valvontaohjelma siirtää toimintaoikeuden ohjausohjelmalle (job control), joka alustaa seuraavan ajon tai ajoryhmän.

Valvontaohjelmaan kuuluu myös muistin listausrutiini, joka tulostaa, ajon päättyessä virheellisesti, sekä yleisrekisterien että keskusmuistin sisällön rivikirjoittimella.

Osa valvontaohjelman rutiineista ladataan keskusmuistiin systeemin alustuksen yhteydessä. Nämä rutiinit pysyvät jatkuvasti muistissa kulloinkin suoritettavasta työstä tai kohdattavasta tilanteesta riippumatta. Edellisten lisäksi kuuluu valvontaohjelmaan joukko rutiineja, jotka haetaan keskusmuistiin käyttöjärjestelmän kirjastosta vain silloin kun niitä tarvitaan. Näistä käytetään nimitystä transientit rutiinit. Kaikki transientit rutiinit käyttävät keskusmuistissa samaa aluetta, ns. yhteisaluetta (transient area), mikä merkitsee sitä, että vain yksi tällainen rutiini kerrallaan voi olla muistissa. Etuna tällä järjestelyllä on muistitilan pieni käyttö huolimatta valvontaohjelman lukuisista toiminnoista.

Ohjausohjelma
(Job Control)

Ohjausohjelma on keskusmuistissa vain ajojen ja ajoryhmien välillä ja sen päätehtävänä on toteutettavien ohjelmien alustus. Ohjausohjelman latauksen suorittaa tarvittaessa valvontaohjelma.

Looginen IOCS
(Syöttö/tulostustoimintojen ohjausjärjestelmä)

IBM toimittaa käyttöjärjestelmän yhteydessä laajan valikoiman makrokäskyjä tiedostojen luontia, käsittelyä ja ylläpitoa varten. Tiedostoa kuvaavien makrokäskyjen avulla kehitetään käyttöohjelman tarvitsemat rutiinit ja muut tiedot. (Jos käyttöjärjestelmänä on BOS, mainitut makrot on sijoitettava muun ohjelman eteen, jolloin niiden perusteella

kehitetyt rutiinit myös sijoittuvat muistissa valvontaohjelman ja varsinaisen käyttöohjelman väliin).

Näiden määritysmakrojen lisäksi tarvitaan toimintomakroja (imperative macro instructions), jotka aiheuttavat haarautumisen määritysmakron perusteella kehitetyn loogisen rutiinin asianmukaiseen käskyyn (toimintomakron sisällön mukaan).

Loogisen IOCS:n tehtäviä ovat:

- fyysisten syöttö/tulostustoimintapyyntöjen esittäminen valvontaohjelmalle tarvittavien makrokäskyjen avulla. Toimintojen edellyttämät kanavaohjelmat kehitetään määritysmakrojen perusteella.

- tiedoston loogisten tietueiden hoito (so. syöttötiedoston loogisten tietueiden toimittaminen yksi kerrallaan käyttöohjelmalle, tulostustiedoston loogisten tietueiden suhteen taas päinvastoin). Tämä funktio sisältää fyysisten tietueiden, lohkojen, purkamisen sekä tietueiden jaksottamisen. Käsittely on samantapainen sekä kiinteän- että vaihtelevanmittaisia tietueita sisältävissä tiedostoissa. (Loogisella tietueella ymmärretään tiedoston pienintä käsiteltävää tietoyksikköä, fyysinen tietue eli lohko taas on useammista tietueista koostuva tietojono, joka luetaan laitteelta tai kirjoitetaan laitteelle yhtenä kokonaisuutena).

- kahden puskurialueen vuorottaisesta käytöstä huolehtiminen. Tällä tavoin voidaan syöttö/tulostustoiminnan kanssa suorittaa käsittelytoimintoja.

- tiedoston ja taltion loppuun liittyvien tilanteiden käsittely.

- tiedostojen luontiin ja ylläpitoon liittyvien järjestelytehtävien hoitaminen. Näihin sisältyvät uusien tietueiden lisääminen tiedostoon, poistot tiedostosta sekä tiedostoon mahdollisesti liittyvien taulukoiden luonti ja ylläpito.

Nimiöiden käsittely

Levystö- ja magneettinauhanimiöiden käsittelyrutiinit ovat tärkeä osa käyttöjärjestelmää, jotta voitaisiin varmistua ensiksikin siitä, että käytettävä tiedosto on oikea ja (jos tiedosto ulottuu esimerkiksi kahdelle tai useammalle taltiolle) että käsittelyjärjestys on oikea. Toiseksi voidaan varmistua siitä että tulostukseen käytettävät taltiot eivät sisällä sellaista tietoa, jota ei vielä saa hävittää. Jälkimmäinen seikka, joka nimiöiden tarkistuksessa saadaan selville, on oikeastaan vielä tärkeämpi kuin edellinen. Vasta kaikkien tarkistusten jälkeen voidaan ko. taltio ottaa käsiteltäväksi (edellyttäen, ettei tarkistuksessa ilmennyt mitään tätä estävää seikkaa).

Nimiöiden tarkistusrutiinit ovat valvontaohjelman transientteja rutiineja, jotka tallennetaan kirjastoon käyttöjärjestelmän luonnin yhteydessä. Rutiinien toteutus tapahtuu yhteisalueella, jolle ne ladataan käyttöohjelman käskyjen (OPEN ja CLOSE makrokäskyt) mukaan. TOS/360 sisältää vain magneettinauhojen nimiörutiinit, BOS/360 ja DOS/360 sisältävät myös levystöjen nimiörutiinit.

OS/360 OHJAUSOHJELMISTO

Kuten muissakin käyttöjärjestelmissä, on OS/360:ssäkin järjestelmän alustustoimintoja varten erillinen ohjelma (IPL Loader).

Tiedon hallinta
(Data Management)

Tämä järjestelmä hoitaa kaikki syöttö- ja tulostuslaitteiden käsittelyyn liittyvät toiminnot. Sen vaikutus tuntuu luonnollisesti parhaiten sellaisten hieman monimutkaisempien laitteiden kohdalla kuten magneettinauhat, levy- ja rumpumuistilaitteet sekä tietoliikennelaitteet. Paitsi 'normaaleja' tiedostoja, järjestelmän avulla pystytään käsittelemään ohjelmia, ts. luomaan ohjelmakirjastoja, noutamaan ohjelmia kirjastosta jne. Menetelmä, jolla ohjelmoija kutsuu tietoja ohjelmaansa, on olennaisesti sama tiedon tyypistä riippumatta.

Eräitä tiedonhallintajärjestelmän lisätoimintoja ovat:

- Tilan varaus poimintamuistilaitteilta tiedostoja varten

- tiedostojen automaattinen paikantaminen

- tiedostojen suojaus, johon sisältyvät varmistus siitä, ettei voimassaolevaa tiedostoa päästä vahingossa tuhoamaan, samaan tietueeseen kohdistuvien samanaikaisten päivitysyritysten

estäminen jne. Eräänä lisämahdollisuutena voidaan mainita luottamuksellisia tietoja sisältävien tiedostojen suojaaminen ns. tunnussanan (password) avulla. (Esim. näin voidaan varmistua siitä, ettei kukaan pääse ilman valtuuksia käsiksi palkkatietoihin).

Työn hallinta
(Job Management)

Tämä ohjausjärjestelmän osa hoitaa koneeseen tulevien töiden (job) suorittamiseen liittyvien ajoitus- ym. järjestelytehtävät nippuajoympäristössä. Toiminta voi tapahtua kahdella tavalla: peräkkäin (jolloin töillä ei siis ole erityistä arvojärjestystä vaan ne suoritetaan siinä järjestyksessä, jossa ne tulevat koneeseen systeemin syöttölaitteelta) tai prioriteettijärjestelmän mukaan (jolloin töiden suoritus tapahtuu käyttäjän määrittämässä 'tärkeysjärjestyksessä').

Tehtävän hallinta
(Task Management)

Työ (job) saattaa koostua yhdestä tai useammasta työvaiheesta (job step) eli ohjelmasta. Työvaiheesta käytetään nimitystä tehtävä (task). Järjestelmän kannalta tehtävä 'syntyy' silloin kun toteutettavalle ohjelmalle on varattu sen tarvitsemat syöttö- ja tulostuslaitteet ja muut resurssit. Työn hallintajärjestelmä (ks. edellinen kappale) hoitaa siis ensin koko työhön liittyvät järjestelytehtävät ja sen jälkeen luovuttaa 'vastuun' tehtävien hallintaohjelmalle, joka puolestaan hoitaa tehtävien (=ohjelmien) toteuttamiseen liittyvät keskeytysten käsittelyn, ohjelmien haun kirjastosta, muistin varauksen ym. tehtävät. Tämä ohjausohjelma pystyy käsittelemään samanaikaisesti yhtä tai useampaa tehtävää (moniajojärjestelmässä tehtävien lukumäärä voi vielä olla joko kiinteä tai vaihteleva) riippuen käytettävästä OS/360 muunnelmasta. Jos useat ohjelmat samanaikaisesti 'kilpailevat' keskusyksikön käytöstä, ratkaisee tehtävien hallintaohjelma, mille tehtävälle toimintaoikeus kuuluu.

TYÖOHJELMAT

Kuhunkin käyttöjärjestelmään kuuluu ohjausjärjestelmän tai -ohjelman lisäksi joukko muita ohjelmia, joita yksinkertaisuuden vuoksi nimitetään tässä työohjelmiksi (processing programs). Työohjelmiin kuuluvat käännösohjelmat, ns. palveluohjelmat sekä installaation omat käyttöohjelmat.

Kääntäjät

Kääntäjä on ohjelma, jonka syötteenä on jollakin ohjelmointikielellä kirjoitettu lausejoukko (lähtöohjelma) ja tulosteena vastaava konekielinen käyttöohjelma.

BOS sisältää Systeemin/360 Assembler-kielen, johon kuuluu täydellinen valikoima makrokäskyjä. Assembler on koneenläheinen, kaikkiin Systeemin/360 malleihin soveltuva symbolikieli. Toinen BOSin kieli on RPG (Report Program Generator), lähinnä raportti- ja tiedostojen ylläpito-ohjelmia varten tarkoitettu tehtävänläheinen määrittelykieli.

DOS ja TOS sisältävät edellisten lisäksi COBOL-, FORTRAN- ja PL/I-kielet. OS, joka on laajin mainituista käyttöjärjestelmistä, sisältää vielä ALGOL-kielen.

Systeemin/360 ohjelmointikielet ovat ylöspäin yhteensopivia (lukuunottamatta Assembler-kieltä, jossa makrokäskyt ovat osittain erilaisia eri käyttöjärjestelmissä). Tämä merkitsee sitä, että DOSin (tai TOSin) FORTRAN-kielellä kirjoitettu lähtöohjelma sopii OSin FORTRAN-kääntäjälle.

Palveluohjelmat
(Service Programs)

Palveluohjelmia ovat linkitysohjelma (linkage editor), kirjastonhoito-ohjelmat sekä systeemin luontiohjelma (vain BOS).

Linkitysohjelma

Linkitysohjelma suorittaa erikseen käännettyjen ohjelmien yhdistämisen, sijoittaa ohjelmat (halutulla tavalla) keskusmuistiin, ratkaisee ohjelmiin sisältyvät ulkoiset viitteet sekä tarvittaessa hakee ohjelmamoduuleita myös käyttöjärjestelmän kirjastosta (välikirjastosta).

Linkitysohjelma suorittaa kaikkien ohjelmien esikäsittelyn ja tallentaa toteuttamiskelpoisen tulostusmoduulin työtiedostoon levymuistille (DOS) tai magneettinauhalle (TOS) seuraavaa toimintaa varten, mikä voi olla ohjelman välitön toteutus tai sen tallentaminen pysyvästi systeemin käyttö-

kirjastoon. Käyttökirjastoon tallennetut ohjelmat voidaan toteuttaa ohjauskorttien määritysten mukaan eivätkä ne enää tarvitse linkitysohjelman käsittelyä.

Kirjastonhoito-ohjelmat
(BOS, DOS, TOS)

Kirjastonhoito-ohjelmien tehtävänä on kirjastojen ylläpito (poistot, lisäykset, kopiointi jne.). Näihin kuuluvat myös rutiinit, joiden avulla kirjastoon tallennettuja ohjelmia (tai kirjastojen hakemistoja) voidaan tulostaa reikäkortinlävistimellä tai rivikirjoittimella. Kirjastoja on kolme, ne sijaitsevat kaikki levystöllä tai magneettinauhalla.

1. Käyttökirjasto (Core Image Library).
   Kirjasto, jonne voidaan tallentaa kaikki valmiit (=toteutuskelpoiset) ohjelmat. Ohjelmien lataamisen muistiin suorittaa valvontaohjelmaan kuuluva systeemin latausrutiini. (Tämä koskee niin IBM:n toimittamia kuin installaation omiakin ohjelmia.)

2. Lähtökirjasto (Source Statement Library).
   Tämä kirjasto sisältää sekä IBM:n toimittamat että installaation omat lähtökieliset moduulit 'kirjat' (esim. makrokäskyt). Kirja on lähtökirjastoon tallennettu (kirjastoitu) nimetty kokoelma lähtökielisiä käskyjä tai lauseita.

   Kirjat tallennetaan tilan säästämiseksi (alkuperäisestä 80 tavusta) tiivistettyyn muotoon kirjaston sijaintitaltioon. Tiivistetty ja lyhennetty muoto nopeuttaa myös hakua. Kirjastosta voidaan poistaa tai lisätä kirjastoon täydellisiä kirjoja, mutta ei yksityisiä tietueita. Kirjoja voidaan liittää käännöksen yhteydessä Assembler- tai COBOL-kielisiin lähtöohjelmiin.

   BOS/360 ei sisällä täydellistä lähtökirjastoa, vaan tätä vastaa makrokirjasto, johon voidaan siis tallentaa ainoastaan Assembler-kielisiä (IBM:n toimittamia tai käyttäjän omia) makrokäskyjä.

3. Välikirjasto (Relocatable Library)
   Tähän kirjastoon voidaan tallentaa ns. välimoduuleja (käännösohjelmien tuloksia), joista myöhemmin saadaan linkitysohjelman avulla valmiita käyttöohjelmia.

Systeemin luontiohjelma
(vain BOS/360)

Tämä on itsenäinen, reikäkorteilla oleva ohjelma, joka sisältää oman alustusrutiininsa (IPL) sekä valvonta- ja ohjausohjelmansa. Sen avulla voidaan luoda levystölle käyttöjärjestelmä reikäkorteilla olevista komponenteista. Lähinnä se on tarkoitettu minimisysteemien luomiseksi erikoissovellutuksia varten. Jos laitteistoon kuuluu kaksi levy-yksikköä, voidaan luontiohjelman sijasta käyttää normaaliin käyttöjärjestelmään kuuluvia kirjastonhoito-ohjelmia.

Lajittelu/yhdistelyohjelmat
(BOS, TOS, DOS)

Lajitteluohjelmien avulla voidaan lajitella umpimähkäisessä järjestyksessä olevat tiedostot yhdeksi järjestyksessä olevaksi tiedostoksi. Yhdistelyohjelman avulla taas voidaan järjestyksessä olevat tiedostot yhdistää yhdeksi (samassa järjestyksessä olevaksi) tiedostoksi. Kunkin tietueen ohjaustieto voi koostua 12 kentästä. Tietueiden lajittelu tai yhdistely voi tapahtua nousevaan tai laskevaan järjestykseen. Nouseva tai laskeva järjestys voidaan määritellä kullekin ohjauskentälle erikseen. Yhdistelyajossa tulostustiedoston järjestyksen on oltava sama kuin syöttötiedosto(je)n.

Rakenteeltaan lajittelu/yhdistelyohjelma on joukko välikirjastossa sijaitsevia moduuleja, joista ajon yhteydessä kootaan kutakin tapausta varten oma ohjelmansa. Käyttäjä määrittelee lajittelu- tai ohjauskentät, kenttien muodon, lajittelujärjestyksen jne. parametrikorteilla. Näiden tietojen mukaan koottu lajitteluohjelma tallennetaan (tilapäisesti) käyttökirjastoon. Ohjelman toteutus tapahtuu kirjastosta vaiheittain (ns. overlay-rakenne).

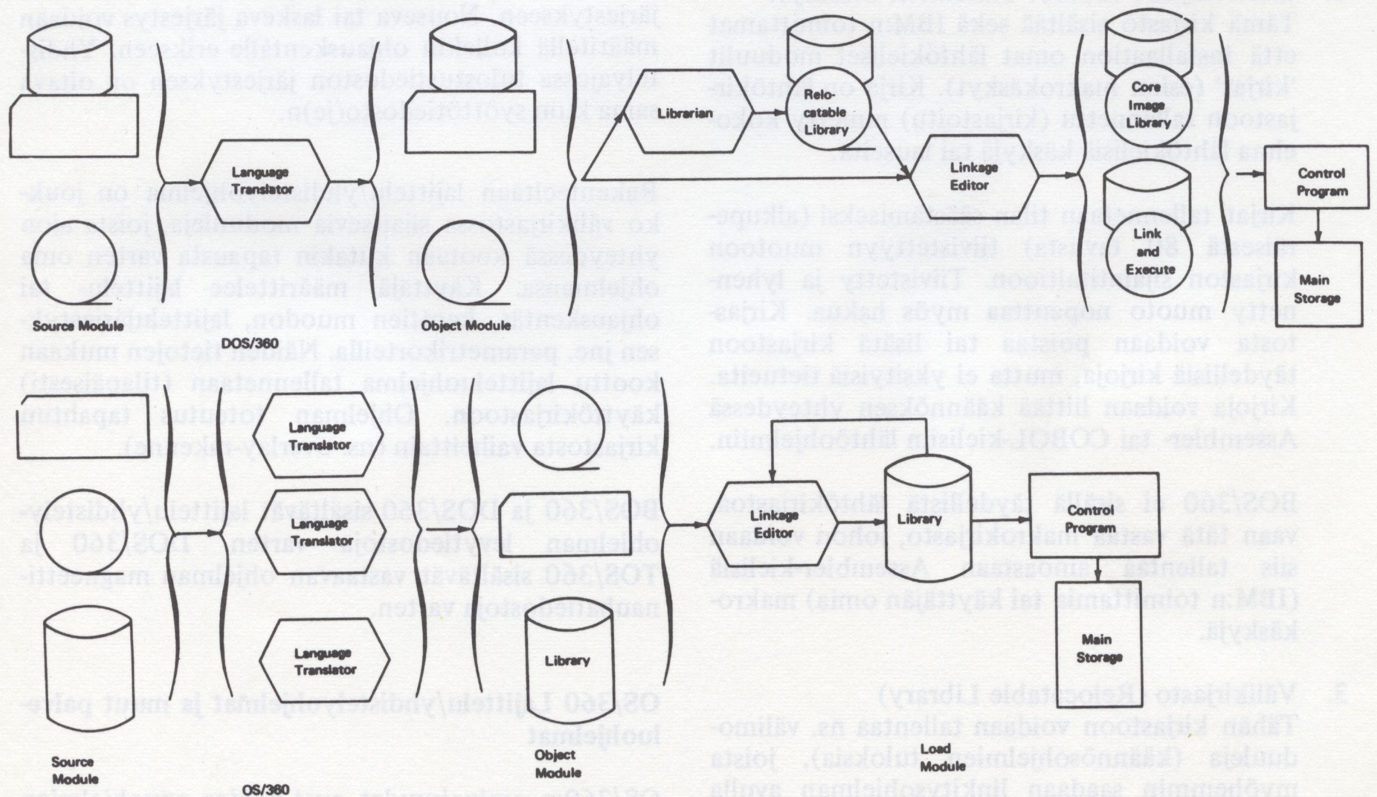
BOS/360 ja DOS/360 sisältävät lajittelu/yhdistelyohjelman levytiedostoja varten. DOS/360 ja TOS/360 sisältävät vastaavan ohjelman magneettinauhatiedostoja varten.

OS/360 Lajittelu/yhdistelyohjelmat ja muut palveluohjelmat

OS/360:n ominaisuudet ovat näiden apuohjelmien suhteen suurin piitein samat kuin DOS/360:n.

Käyttömahdollisuuksiltaan ohjelmat ovat kuitenkin laajempia ja sisältävät vähemmän rajoituksia kuin DOS/360:n vastaavat ohjelmat. DOS/360 lajittelu/yhdistelyohjelmat sisältävät esimerkiksi tiedostojen kokoon ja lukumäärään liittyviä rajoituksia, kun taas OS/360 ohjelmat ovat huomattavasti 'vapaamielisempiä' tässä suhteessa. Lisäksi OS/360 lajittelu/yhdistelyohjelmat ovat monipuolisempia ja nopeampia.

OS/360 ei sisällä kirjastonhoito-ohjelmia samassa mielessä kuin DOS, koska OS käsittelee kaikkia tiedostoja (ohjelmat näihin mukaanluettuina) samalla tavoin; tiedostojen ylläpitoa varten on kuitenkin olemassa joukko apuohjelmia. Lisäksi OS/360:n käyttäjällä on mahdollisuus omilla ohjelmillaan (tai omilla makrokäskyillään) suorittaa tiedostojen ylläpitotehtäviä. Miten eri käyttöjärjestelmät käsittelevät ohjelmia (lähtöohjelmasta alkaen) selviää kuvasta 129.



Kuva 129. Käyttöohjelman eri vaiheet käyttöjärjestelmissä DOS/360 ja OS/360.

APUOHJELMAT

BOS/360 sisältää yksitoista apuohjelmaa tiedostojen siirtoa varten laitteelta tai tallennusvälineeltä toiselle:

- nauhalta nauhalle
- nauhalta levylle
- nauhalta korteille
- nauhalta rivikirjoittimelle
- levyltä nauhalle
- levyltä levylle
- levyltä korteille
- levyltä rivikirjoittimelle
- korteilta nauhalle
- korteilta levylle
- korteilta rivikirjoittimelle ja/tai korteille

Muita BOS/360:n apuohjelmia ovat

- levystön tyhjäys. Tyhjättävä alue voi olla yhdestä urasta koko levystöön asti.

- magneettinauhojen vertailuohjelma, jonka avulla voidaan tutkia kahden tai useamman nauhatiedoston identtisyys.

TOS/360 sisältää samat apuohjelmat kuin BOS/360 lukuunottamatta sellaisia, joihin liittyy levystöjen käsittely. DOS/360 sisältää kaikki BOS/360:n apuohjelmat sekä lisäksi seuraavat kuusi ohjelmaa 2321 kennomuistia varten:

- nauhalta kennomuistiin
- levyltä kennomuistiin
- kennomuistilta nauhalle
- kennomuistilta levylle
- kennomuistilta kennomuistille
- kennomuistilta rivikirjoittimelle.

BOS/360, TOS/360 ja DOS/360 apuohjelmat ovat laitteesta riippuvaisia. OS/360 apuohjelmat ovat laitteesta riippumattomia, koska niissä viitataan pikemminkin tiedostoihin kuin laitteisiin. Muuten niiden toiminta on samankaltaista kuin DOS/360 apuohjelmienkin. OS/360:n puitteissa voidaan käyttää seuraavia poimintamuistilaitteita:

- 2301 rumpumuisti
- 2302 levymuisti
- 2303 rumpumuisti
- 2311 levymuisti
- 2314 levylaitteisto
- 2321 kennomuisti

Standardien laatimista ja kehittämistä ovat huomattavasti helpottaneet sellaiset järjestöt kuin SHARE, GUIDE ja COMMON. Näiden järjestöjen jäsenet voivat kokouksissaan vaihtaa kokemuksia sekä kirjastojen avulla myös ohjelmia.

Ohjelmien testaus ja virheiden etsintä

Ohjemoija, jolla on käytettävissään erityisiä testauksessa tarvittavia apuohjelmia, voi näiden avulla säästää huomattavasti vaivojaan ohjelmavirheiden etsinnässä. Systeemin/360 käyttöjärjestelmiin kuuluu tällaisia rutiineja. BOS/360 ja TOS/360 ja DOS/360 käyttöjärjestelmissä testauksen apuväline on nimeltään Autotest. OS/360:ssä se on nimeltään Testran. Kummankin toimintaperiaate on samanlainen: ohjelmoija voi haluamissaan ohjelman kohdissa käyttää testiohjelman palveluksia hyväkseen. Näitä palveluksia ovat mm. muistin vedokset (dump), rekisterien sisällön tulostus, 'kirjanpito' ohjelman toiminnasta (erityisesti haarautumiskäskyt), alirutiinien kutsumiset jne. Testausohjelmat pystyvät myös tutkimaan erilaisia käyttöohjelman toteutuksesta johtuvia tilanteita ja näiden tulosten perusteella joko suorittamaan tai jättämään suorittamatta ohjelmoijan määrittämiä muistivedospyyntöjä tms.

KÄYTTÖOHJELMAT

Tietokoneiden käyttäjät laativat ohjelmia ongelmiensa ratkaisemiseksi. Jotkut ohjelmista saattavat olla hyödyllisiä myös muille, joilla on vastaavanlaisia ongelmia. Muiden laatimien rutiinien liittäminen installaation omiin ohjelmiin tekee tietyt ohjelmien laatimiseen liittyvät menettelytavat erityisen tärkeiksi. Ohjelmien vaihto edellyttää toisin sanoen standardeja.

Standardien kehittämistä ja vakiintumista käyttöön ovat helpottaneet IBM:n asiakkaiden itsenäiset yhdistykset kuten COMMON, GUIDE ja SHARE. Yhdistysten tarkoituksena on jäsenten keskinäisen tietojen ja ohjelmien vaihdon lisääminen yhteistyössä IBM:n kanssa. Järjestöjen erilaisia projekteja varten asettamat toimikunnat julkaisevat säännöllisesti töittensä raportteja, jotka jaetaan kaikille jäsenille.

Lisätietoja näistä järjestöistä on saatavissa paikallisilta IBM:n edustajilta.

KAUKOKÄSITTELY

Käyttöjärjestelmät DOS/360 ja OS/360 sisältävät myös tietojen kaukokäsittelymahdollisuuden. Tässä suhteessa molemmat käyttöjärjestelmät ovat pohjimmiltaan samanlaisia. Kummassakin on mahdollisuuksia hyvinkin monenlaisiin kaukokäsittelysovellutuksiin, joista esimerkkejä ovat:

- *Sanomanvälitys (Message Switching)*
  Tällä tarkoitetaan jostakin päätteestä (terminal) saatujen sanomien edelleen lähettämistä jollekin toiselle (tai useammille kuin yhdelle) päätteelle.

- *Töiden kaukokäsittely (Remote job processing)*
  Tällaisessa sovellutuksessa pääte toimii syöttö-

laitteena, josta työt (ajot, ohjelmakäännökset, tms.) luetaan systeemiin toteutettavaksi ohjausohjelman valvonnassa.

- *Kyselyt ja tapahtumatietojen käsittely*
  Esimerkiksi pankkisovellutuksena käytössä. Haarakonttoreihin sijoitetuista päätteistä voidaan suorittaa tiliasemaa koskevia kyselyjä ja/tai suorittaa tilin päivitys otto- tai jättötapahtuman perusteella välittömästi. Varsinaisen käsittelypuolen tällaisessa sovellutuksessa hoitaa käyttäjän oma ohjelma.

- *Tietojen keräily (Data Collection)*
  Sovellutuksena periaatteessa hieman edellisen kaltainen. Tässä tosin tiedot vain kootaan tietokoneelle sopivaan muotoon ja taltioidaan myöhempää käsittelyä varten.

Tietoliikennettä varten on käytettävissä kolme saantimenetelmää, nimittäin BTAM (Basic Telecommunication Access Method), QTAM (Queued Telecommunication Access Method) sekä STRAM (Synchronous Transmit Receive Access Method).

BTAM sisältää ohjaustoiminnot, jotka liittyvät päätteisiin kohdistuviin kiertokyselyihin (polling) ja kutsuihin (addressing) sekä sanomien lähetykseen ja vastaanottoon. Käyttöohjelman huoleksi sen sijaan jäävät:

1. Kiertokyselyjen ja kutsujen alustustoimet (makrokäskyjen avulla).

2. Sanomien ohjaaminen haluttuihin paikkoihin (muille päätteille ja ohjelman käsiteltäväksi).

3. Virheiden tarkistus ja käsittely.

4. Koodien muunto, nimiöiden tarkistus, sanomien rekisteröinti ja sanomajonojen käsittely.
5. Sanomien varsinaisen sisällön käsittely.

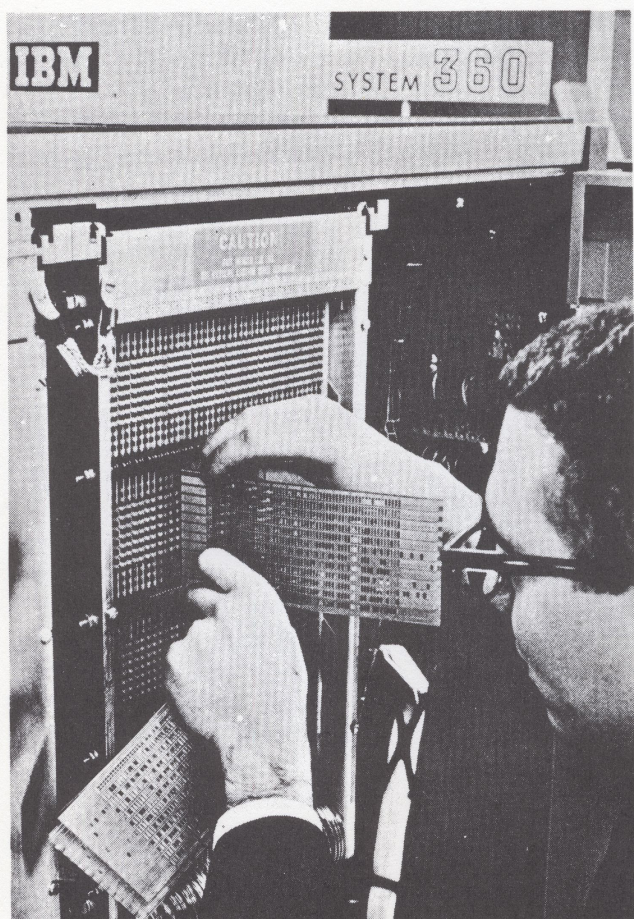
QTAM sisältää kaikki edellä mainitut funktiot paitsi tietenkin sanomien sisällön käsittelyä.

Sekä BTAM että QTAM soveltuvat päätteille, jotka ovat start/stop-tyyppisiä (esim. IBM 1050-järjestelmä). Näiden tietojen siirtonopeus (lähetysnopeus) on yleensä noin 15 merkkiä sekunnissa.

STRAM sisältää synkronisilla kaukosiirtolaitteilla tapahtuvaa sanomien lähetystä ja vastaanottoa varten tarvittavat makrokäskyt ja ohjausrutiinit. STRAM sisältää seuraavat toiminnot:

1. Linjojen ohjaus.
2. Laitteiston määritysmakrot.
3. Koodien muunto.
4. Tietojen siirto.
5. Virheiden käsittely.

Edellisten 'tietokonesukupolvien' aikana uuden tietokoneen hankinta aiheutti yleensä myös vanhaa konetta varten laadittujen ohjelmien uudelleen kirjoittamisen. Ohjelmien siirto koneesta toiseen oli siten paitsi hidasta myös kallista. Jotta vanhan koneen korvaaminen Systeemillä/360 kävisi ohjelmoinnin kannalta helpommin, on IBM kehittänyt juuri siirtymävaihetta silmälläpitäen tarvittavia ohjelmia sekä koneisiin tarvittavia lisälaitteita.



Siirrettäessä ohjelmia Systeemin/360 käyttöön on huomattava merkitys tällä erikoiskortilla, jota huoltoteknikko on sovittamassa mallin 30 lukumuistiyksikköön. Lukumuisti sisältää joukon tällaisia käskyjä sisältäviä kortteja, joiden avulla esim. IBM 1401-tietokonetta varten kirjoitettuja ohjelmia voidaan muutoksitta toteuttaa Systeemillä/360.

Kuva 130. Lukumuistiyksikkö ja lukumuistikortti (ROS-kortti)

Systeemin/360 yhteensopivuuslaitteiden avulla voidaan tietyillä konemalleilla toteuttaa muita koneita varten laadittuja ohjelmia. Ohjelmiin ei tarvitse tehdä mitään muutoksia edellyttäen, että ne noudattavat ko. tietokonejärjestelmän (esim. 1401) normaaleja koodausstandardeja. Yhteensopivuus perustuu tässä tapauksssa ns. lukumuistiin (Read-Only Storage, ROS).

Lukumuisti on keskusyksikköön rakennettu toimintajärjestelmä, jossa toiminta-alkiot ovat varsinaisten konekäskyjen osia (samalla tavoin kuin konekäskyt puolestaan ovat esim. FORTRAN-kielisen lauseen osia). Näistä toiminta-alkioista koostuvaa toimintasarjaa nimitetään mikro-ohjelmaksi. Sen sijaan että mikro-ohjelmassa toteutettaisiin käskyjä, siinä ohjataan tietovirran kulkua avaamalla ja sulkemalla 'elektroniportteja'. Kuvassa 130 on nähtävissä ns. ROS-kortti. Kukin lävistetty kortin rivi ohjaa yhden konekielisen käskyn toteuttamiseen tarvittavan tiedon kulkua. Useimpiin Systeemin/360 keskusyksiköihin on rakennettu lukumuisti systeemin omia käskyjä varten: yhteensopivuus vaatii aivan oman lukumuistinsa. 1401/1440/1460 tai 1620 ohjelmia varten tarvittava 1401 tai 1620 lukumuisti muuntaa ko. konejärjestelmän käskyt joukoksi mikrokäskyjä, jotka Systeemi/360 pystyy toteuttamaan.

Emuloinnissa yhdistetään yhteensopivuuslaite (ts. lukumuisti eli ROS) ja muistissa oleva ns. emulaattoriohjelma, joiden avulla myöskin muistissa olevan 'vieraan' ohjelman toteutus tapahtuu.

Toteutus tapahtuu yhteensopivuuslaitteen valvonnassa niin kauan kunnes se toteaa, ettei pysty jotakin toimintoa käsittelemään. Tässä vaiheessa toimintaoikeus siirtyy emulaattoriohjelmalle, jossa kyseisen operaation toteutus tapahtuu Systeemi/360-tilassa. Toiminnon päätyttyä siirtyy toimintaoikeus jälleen yhteensopivuuslaitteelle seuraavan käskyn käsittelyä varten. Emulaattoriohjelmia on saatavissa useita, mm:

Alkuperäinen kone	Tarvittava Systeemin/360 malli
1620	malli 30
1401/1440/1460/1410/7010	malli 40
7070/7074/1410/7010	malli 50
705/7080/709/7090/7094/7040/7044	malli 65

Pari emulointiin liittyvää huomionarvoista seikkaa ovat:

- ohjelman vaatima muistitila. (Emulaattorikoneessa tarvittava muistitila on aina suurempi kuin alkuperäisessä koneessa).
- laitteisiin ja käskykoodeihin mahdollisesti liittyvät rajoitukset.

Vaikka vanhoja ohjelmia voidaankin emulaattorien ja yhteensopivuuslaitteiden avulla ajaa Systeemillä/360 vähintään yhtä tehokkaasti kuin entisillä koneilla, ei tällä järjestelyllä kuitenkaan pystytä täysin käyttämään hyväksi Systeemin/360 kaikkia mahdollisuuksia ja kykyjä.

Tietojenkäsittelyproseduuriin kuuluu kaksi olennaista toiminta-aluetta: tulosten aikaansaaminen ja menettelyn valvonta. Täydellinen valvontajärjestelmä sisältää paljon enemmän kuin vain tietokoneen suorittamien töiden laadun tarkkailu. Valvonnan on katettava koko sovellutus ja otettava huomioon sen merkitys yritykselle.

Valvontajärjestelmän täytyy sisältää sopivat menetelmät, joilla pystytään valvomaan tietojenkäsittelyjärjestelmään saapuvia ja sieltä lähteviä tietoja ja varmistamaan, että vaadittuihin tuloksiin myös sisältyy kaikki järjestelmään saapunut informaatio. Lisäksi on oltava menetelmä, jolla jonkin tiedon puuttumisen tai toistumisen aiheuttamat virhetilanteet saadaan kiinni tarvitsematta käydä läpi koko käsittelyproseduuria uudelleen.

Valvontamenetelmät poikkeavat toisistaan jonkin verran kaupallisissa ja toisaalta teknisissä, tilastollisissa, tieteellisissä ja matemaattisissa sovellutuksissa, vaikkakin kumpaakin tyyppiä saattaa olla käytössä samassa installaatiossa. Tieteellisissä sovellutuksissa valvonta kohdistuu pääasiassa vain laskennan tarkkuuteen ja koska tiedon valvonnan on myös oltava tiukkaa, ovat jälkitarkkailulle asetettavat vaatimukset yleensä varsin yksinkertaiset taikka niitä ei ole lainkaan.

Toisaalta liikeyrityksen tietomateriaali on oikeastaan osakkeenomistajien omaisuutta ja sen täytyy tämän vuoksi olla sekä ulkonaisesti että sisäisesti tarkistettavissa. Liikeyrityksen asiakirjat on myös suojattava niin, että niitä ei pystytä käyttämään väärin. Erityisvaatimuksia niille saattavat asettaa verolait, julkinen hallinto yleensä taikka paikalliset tavat tai olosuhteet. Monissa yrityksissä koko kirjanpitojärjestelmä vaikuttaa suoraan yrityksen ja sen asiakkaiden välisiin suhteisiin. Tämä taas edellyttää tiedostojen järjestämistä sellaisiksi, että niistä löytyvät vastaukset tiliasemaa, varastotilannetta, tms. koskeviin kyselyihin aina yrityksen luonteesta riippuen.

Seuraava esitys koskettelee lähinnä kaupallisia sovellutuksia, mutta yleisiä periaatteita voidaan kuitenkin noudattaa missä tahansa sovellutuksessa. Valvontajärjestelmän tarkoituksena on:

TIETOJENKÄSITTELYMENETELMIEN VALVONTA

1. Varmistaa tietokonekäsittelyyn saapuvien tietojen oikeellisuus.

2. Tarkistaa käsittely (ihmisten suorittama) ennen tietojen saapumista konekäsittelyyn.

3. Järjestellä tiedot sellaiseen muotoon, missä ne taloudellisimmin soveltuvat tietokoneen käyttöön.

4. Järjestää koko prosessin eri vaiheiden tarkistusmenetelmät siten, että mahdolliset virheet pystytään paikantamaan mahdollisimman lyhyessä ajassa.

5. Varmistaa se, että verotusta tai muuta julkista hallintoa varten tarkoitettu informaatio on lakien mukaista.

6. Suojella yritystä väärinkäytöksiltä, jotka vaikuttavat sen taloudelliseen asemaan tai maineeseen.

VALVONTARYHMÄT

Tietokonekeskus on yleensä palveluyksikkö. Se vastaanottaa käsiteltävää informaatiota (reikäkorteille tai muulle tallennusvälineelle rekisteröitynä) yrityksen eri osastoilta tai myös yrityksen ulkopuolelta, suorittaa tarvittavan käsittelyn ja valmistaa tarvittavat raportit. Keskus ei yleensä itse synnytä informaatiota, sen tehtävänä on vain sille lähetettyjen tietojen käsittely. Näin ollen voidaan keskuksen henkilökuntaa valvoa ja estää väärinkäytökset ja epätäsmällisyydet tietojen käsittelyssä. Valvontatehtävä kuuluu yleensä erityiselle ryhmälle, joka on perustettu keskukseen tai keskuksen ulkopuoliselle, muuten sitä lähellä toimivalle ryhmälle.

Palkkatietojen valvonta

Esimerkiksi voidaan ottaa jokin suuri yhtiö, jonka palkanlaskenta tapahtuu tietokoneilla. Yhtiössä on erityinen palkkakonttori. Kaikki palkkatietoja

koskevat muutokset - ennakonpidätysperusteiden muutokset, uudet työntekijät, työsuhteiden päättymiset jne. - kulkevat tämän konttorin kautta. Muutokset ilmoitetaan erityisillä lomakkeilla, joista alkuperäinen kappale toimitetaan ko. osaston hyväksymismerkinnällä varustettuna palkkakonttoriin jäljennöksen jäädessä osastolle.

Palkkakonttorissa ryhmitellään muutoslomakkeet sen mukaan mitä tietoja ne koskevat. Lomakkeiden numeerisista kentistä lasketaan tarkistussummat, koskivatpa kentät sitten markkamääriä tai työntekijöiden tunnustietoja, työhönotto- ja työsuhteen päättymisilmoituksissa tarkistussummaan otetaan mukaan myös työntekijöiden lukumäärä. Tämän jälkeen aineistosta lävistetään reikäkortit tietokonetta varten. Ennen käsittelyä suoritetaan myös tarkistuslävistys.

Muutosten kokonaismäärä kootaan viikottain käsittelemällä reikäkortit taulukointikoneella. Saadut tarkistussummat tarkistetaan aikaisemmin laskukoneilla laskettuihin summiin vertaamalla. Tietokoneajossa lasketaan normaalit palkat ja suoritetaan tarvittavat vähennykset. Tulokset lähetetään jälleen palkkakonttoriin, jossa niitä verrataan siellä rekisteröityinä oleviin tietoihin. Tämä valvontajärjestelmä ei siis koske ainoastaan tietojenkäsittelyjärjestelmän toimintaa vaan myös ihmisten suorittamaa työtä ja systeemiin saapuvan informaation keräilyä.

Myyntitietojen valvonta

Menetelmien valvonnan eräänä päätarkoituksena on hyvien asiakassuhteiden säilyttäminen ja edelleen kehittäminen. Eräänä esimerkkinä voidaan mainita yrityksen myymien hyödykkeiden ja palvelusten hintastruktuurin valvonta. Pelkästään tähän tehtävään voidaan perustaa oma valvontaryhmä.

Monissa yrityksissä myyntiosaston vastuulla on hinnoittelu. Vahingoittuneesta tavarasta tai jostakin muusta erityisestä syystä myönnetyt alennukset saattavat kuitenkin aiheuttaa poikkeamia vahvistetusta hintatasosta. Valvontaryhmän tehtävänä tällaisissa tapauksissa on huolehtia siitä, että kaikki poikkeamat tutkitaan ja perustellaan. Poikkeustapausten raportointi voidaan hoitaa tietokoneella.

Tavaran tunnustiedot, tuotenumero, sekä asiakaskoodi syötetään tietokoneeseen samoin kuin erikoinen huomautus mikäli tuotteen hinta poikkeaa normaalista. Tietokone hinnoittelee tavaralähetykset käyttäen tiedossaan olevia hintoja, poikkeustapauksissa noudatetaan määritettyjä erikoisohjeita. Ajon tuloksena syntyy tiedosto, jonka perusteella kirjoitetaan laskut asiakkaille ja jonka perusteella tulostetaan lista niistä lähetyksistä, joissa on käytetty erikoishintoja. Poikkeuslista lähetetään valvontaryhmän tutkittavaksi.

Valvontaryhmällä saattaa olla muitakin samantapaisia tehtäviä. Mitä hinnoitteluun ja laskutukseen tulee, koneellinen menetelmä eroaa huomattavasti käsimenetelmästä. Jos laskutus hoidetaan käsin, on inhimilliset erehdykset otettava huomioon vaikka alkuperäiset hintatiedot olisivatkin oikeat. Laskuttajathan käyttävät päivän tasalla olevaa hinnastoa.

Koneellisessa menetelmässä toisaalta hinnat poimitaan hintatiedostosta siten, että esim. tuotenumeron perusteella etsitään asianomainen tavaratietue. Jos jonkin tuotteen hinta on virheellinen, virhe näkyy jokaisessa ko. tuotetta sisältävässä laskussa. Tämän vuoksi perushintatiedoston sisällön valvonnan on oltava erittäin tiukkaa.

Perustiedoston muutokset voidaan hoitaa omana tietokoneajonaan. Tiedostoon tallennetaan uudet hinnat ja tuloksena saadaan muutostiedosto, josta kopio puolestaan toimitetaan valvontaryhmälle tarkistusta varten. Silloin tällöin on syytä listata koko tuote- ja hintatiedosto. Tiedoston perusteella voidaan myös laatia kätevästi hinnasto mahdollisia käsin suoritettavia toimenpiteitä varten.

Samaan tapaan voidaan kehittää valvontamenetelmä, jolla varmistetaan, että tavaran toimitus tapahtuu vain sellaisille asiakkaille, joiden tiliasema täyttää tietyt vaatimukset. Tämä valvonta voidaan parhaiten hoitaa asiakastiedoston avulla, josta lähetyksiä varten joudutaan etsimään (asiakkaan numeron mukaan) nimi- ja osoitetiedot. Samassa yhteydessä on helppo tarkistaa sopiiko asiakkaan tilaus hänelle määrättyihin luottorajoihin. Asia voidaan hoitaa myös siten, että luottokaupat jätetään luotto-osaston hyväksyttäviksi. Joillakin erikoisaloilla näin voidaan myös valvoa, että tavara toimitetaan vain valtuutetuille kauppiaille (so. asiakkaille), koska nimen puuttuminen tiedostosta automaattisesti estää sekä laskun kirjoittamisen että tavaran toimittamisen.

SYSTEEMIN TARKISTUKSET

Systeemin tarkistuksilla tarkoitetaan tietokoneella hoidettavan proseduurin valvontaa. Niillä pyritään varmistamaan, että kaikki koneeseen saapuvat ja sieltä lähtevät tiedot ovat täydellisiä ja tarkkoja.

Systeemin tarkistuksiin saattaa kuulua varmistuminen siitä, että kaikki syöttötietueet todella tulevat käsitellyiksi määrätyn ajan kuluessa ja että virheelliset tai asiaankuulumattomat tietueet tulevat asianmukaisesti poistetuiksi. Systeemin tarkistuksiin saattavat myös kuulua erilaiset laskutoimitusten loogisuus- ja järkevyysvaatimukset. Samoin niihin kuuluvat varsin yleisesti käytetyt ristitarkistukset eli useiden eri teitä koottujen loppusummien avulla suoritetut tarkistukset.

Systeemin tarkistukset vaihtelevat sovellutuksen ja käytettävän laitteiston mukaan. Tarkistuksiin on syytä kiinnittää huomiota jo sovellutuksen suunnitteluvaiheessa, koska niiden liittäminen ohjelmiin käy sitä helpommin mitä aikaisemmassa vaiheessa ne laaditaan. Joissakin tapauksissa voidaan joutua muuttamaan systeemiä sen takia, että saataisiin tarpeeksi tehokkaat tarkistusmenetelmät mukaan. Tämä on yleensä kuitenkin paljon halvempaa kuin systeemin laatiminen ilman mitään tarkistuksia.

Monet kaupalliset sovellutukset edellyttävät hyvin tarkkaa menetelmän ja laskennan valvontaa sekä lisäksi jälkitarkistusmahdollisuutta. Tämä merkitsee sitä, että ohjelmassa on mahdollisimman suuressa määrin voitava käyttää hyväksi tietokoneen mahdollisuuksia ja luotettavuutta. Koko systeemin tehokas toiminta edellyttää myös, että virheen sattuessa sen aiheuttaja pystytään paikantamaan hyvin nopeasti ilman, että koko prosessia joudutaan käymään läpi uudelleen.

Joissakin koneissa sisäänrakennetut tarkistusjärjestelmät tekevät monet yksityiskohtaiset systeemin tarkistukset tarpeettomiksi. Toisissa taas keskusyksikössä tapahtuvaa tiedon käsittelyä ei niin tarkoin valvota, esimerkiksi silloin kun monimutkaiset tarkistusjärjestelmät aiheuttaisivat kustannusten tuntuvan lisäyksen ilman että tarkkuus samassa suhteessa nousisi. Osa tarkistuksista voidaan hoitaa ohjelmointi- ja käyttöjärjestelmien avulla.

Tarkistus on eräs muoto laaduntarkkailua. Jos systeemissä sallitaan tietty virhemäärä, voidaan tarkistuksia vastaavasti vähentää, sama pätee päinvastoin. Kysymys on itse asiassa tarkoituksenmukaisuudesta kunkin systeemin kohdalla. Seuraavassa esityksessä kosketellaan eräitä tyypillisiä tarkistusmenettelyjä.

Tietueiden laskeminen

Tietueiden lukumäärään sisällytetään kaikki tiedoston tietueet. Alkuperäinen lukumäärä saadaan tiedoston luonnin yhteydessä.

Tietueiden lukumäärä tallennetaan tiedoston loppuun (tai alkuun) ja sitä muutetaan tietueita poistettaessa tai lisättäessä. Joka kerta kun tiedostoa käsitellään, tietueet lasketaan uudelleen ja verrataan lukumääriä siihen mikä tiedostoon on tallennettu. Jos lukumäärät ovat yhtäsuuret, voidaan katsoa että kaikki tietueet tulivat käsitellyiksi (tai ettei yhtään tietuetta ole vahingossa hävinnyt).

Tietueiden lukumäärä olisi laskettava myös eräkohtaisesti varsinkin silloin, kun lähtötiedot otetaan käsiteltäviksi ensimmäistä kertaa.

Vaikka tietueiden laskeminen onkin melko hyödyllinen tarkistusmenettely, on vaikea määrittää virheen syytä siinä tapauksessa, että lukumäärät eivät täsmää. Lukumäärien poikkeaminen toisistaan ei auta puuttuvan tietueen paikantamisessa eikä myöskään paljasta mikä tietue mahdollisesti on käsitelty kahteen kertaan. Tämän vuoksi täytyy olla mahdollista tarkistaa tiedosto vertaamalla sitä lähtötietoihin, tiedoston kopioon tai listaan, joka varmasti sisältää tietueiden todellisen lukumäärän jne.

Virheellinen tietueiden lukumäärä saattaa johtua käsittelyn yhteydessä tapahtuneesta konevirheestä, koska magneettinauhalle, levystölle tms. kerran tallennettu tietue ei voi sieltä itsestään hävitä. Jos näin on käynyt, joudutaan tiedoston virheellinen osa (tai koko tiedosto) uusinta-ajossa korjaamaan. Mitä virheelliseen tietueeseen tulee, ilmoittaa käyttöjärjestelmä yleensä virheen syyn.

Tarkistussummat

Tarkistussumma voidaan kerätä esimerkiksi markka- tai kappalemääriä sisältävistä kentistä. Alkuperäinen tarkistussumma syntyy tiedoston luonnin yhteydessä koneellisesti tai käsin laskettuna.

Tarkistussumma voi käsittää koko tiedoston (grand total), tai se voi jakautua useihin välisummiin.

Tiedostoa käsiteltäessä kerätään summat jälleen samoista kentistä ja verrataan tarkistussummaan. Jos summat täsmäävät, voidaan katsoa, että (ainakin siihen saakka) kaikki tietueet on käsitelty oikein.

Tarkistussumma on tehokas menettely silloin kun sen avulla voidaan ennakolta määrittää laskennan tuloksia. Esimerkiksi palkanlaskennassa kaikkien työntekijöiden tekemä tuntimäärä saadaan etukäteen selville tuntilomakkeiden tai kellokorttien perusteella. Tästä luvusta saadaan tarkistussumma palkka-ajon raportteja varten. Tarkistussumma voidaan edelleen jakaa välisummiin esimerkiksi osastoittain. Välisummista kerätyn kokonaissumman on täsmättävä alunperin koottuun summaan.

Tarkistussummat kootaan yleensä ryhmittäin. Loogisen ryhmän voi muodostaa osasto, konttori, tili tms. Tällä tavoin saadaan ryhmäkohtainen tarkistus, joka helpottaa virheiden hakua.

Tarkistusluvut

Ns. koelukuja voidaan käyttää esimerkiksi kertolaskujen tarkistukseen. Koeluvun avulla tarkistetaan sekä systeemi että tietokoneen toiminta. Koeluku on yleensä tietueeseen sisältyvä lisätieto.

Esimerkkinä voidaan ottaa kertolasku, jossa tekijöinä ovat määrä ja yksikkökustannus. Tarkistus perustuu todellisen ja ns. koekustannuksen väliseen suhteeseen. Tätä varten määritetään mielivaltainen luku (Z), joka on suurempi kuin mikään normaali kustannus. (Jos kustannus yleensä on 0,50 markan ja 10,95 markan välillä, voidaan luvuksi Z ottaa 11,00 markkaa). Ns. koekustannus on jäännös, joka saadaan vähentämällä todellinen kustannus Z:sta, eli

Kustannus + koekustannus = Z

Koekustannus on ylimääräinen tieto, joka sisältyy kaikkiin tietueihin. Z taas voi olla ohjelmavakio.

Joka kerran kun kustannus kerrotaan määrällä, kerrotaan myös koekustannus, jolloin käsittelyn yhteydessä saadaan siis kolme tekijää: määrä, määrän kustannuksen tulo ja määrän ja koekustannuksen tulo. Tarkistus tapahtuu näillä luvuilla seuraavan kaavan mukaan:

Σ (Määrä x kustannus) + Σ (määrä x koekustannus) = Σ (määrä x Z)

Yhtälön vasen puoli saadaan laskemalla yhteen mainitut kaksi käsittelyn aikana syntyvää summaa per käsiteltävä tapahtuma. Oikea puoli taas on ajossa kootun kokonaismäärän ja vakion Z tulo. Tällä tarkistusmenettelyllä varmistetaan, että kukin kertolasku on suoritettu oikein. Samankaltaista menettelyä voidaan käyttää hyvinkin monissa sovellutuksissa.

Rajatarkistus

Rajatarkistus tarkoittaa tietueen kenttien ja summien vertaamista tiettyihin ennalta määrättyihin raja-arvoihin, joita tutkittava tieto ei saa ylittää (tai alittaa). Jos esimerkiksi tapahtumakoodien on oltava numeerisia ja arvojen on oltava 0-5, voidaan ohjelmassa tarkistaa, ettei mikään koodi ole suurempi kuin 5.

Toinen tyypillinen rajatarkistus liittyy laskettujen summien järjellisyyteen. Tämä edellyttää, että tiedetään etukäteen joidenkin määrien ja arvojen vaihtelurajat.

Palkanlaskenta sisältää useinkin monia sellaisia rajoituksia, joita voidaan käyttää ohjelmassa suoritettavissa tarkistuksissa. Bruttopalkan yläraja voidaan esimerkiksi määrittää suhteellisen helposti palkkalajin mukaan (tunti-, kuukausi-, kappalepalkka jne.). Tuntiansioiden taas on sovittava tiettyyn skaalaan, samoin viikottaisella tuntimäärällä per työntekijä on tietty ylärajansa.

Rajatarkistusta voidaan käyttää myös taulukkohaussa. Jos tieto on muistissa olevassa taulukossa, laskettu taulukko-osoite voidaan tarkistaa vertaamalla sitä taulukon korkeimpaan osoitteeseen. Tällä menettelyllä saadaan ainakin karkeimmat virheet välittömästi selville.

Monissa matemaattisissa tehtävissä voidaan etukäteen yleensä arvioida lopputulosten arvoalue. Jos tulos sattuu arvioidun arvoalueen ulkopuolelle, voidaan olettaa, että kyseessä on virhe, joka johtuu joko ohjelmasta tai lähtötiedoista. Jos lasketuilla luvuilla on olemassa tietty trendi, huomattavat

poikkeamat tästä saattavat myös kertoa virheen olemassaolosta. Yksinkertaisten rajatarkistusten käyttö, jolla kuitenkin voidaan saada selville monia virheitä, saattaa säästää ohjelmoijalta paljon vaivaa (verrattuna pieniin detaljeihin meneviin tarkistuksiin), koska se ei ohjelmoinnille aseta kovinkaan suuria vaatimuksia.

Ristitarkistus

Ristitarkistusta voidaan käyttää joko ennalta laskettujen tai ajon aikana koottujen tarkistussummien yhteydessä. Esimerkiksi palkka-ajossa lasketaan bruttopalkkojen, verojen, muiden vähennysten ja nettopalkkojen summat. Yleensä summien laskenta tapahtuu osastoittain (tai jonkin muun soveltuvan ryhmityksen mukaan). Jokaisen katkon kohdalla voidaan tarkistaa summien yhtäpitävyys, ts. onko bruttopalkkojen summa yhtäsuuri kuin nettopalkkojen ja vähennysten summa.

Magneettinauha ja levystönimiöt

Magneettinauhakelan alkuun tallennettua tunnustietoa sanotaan alkunimiöksi (header label), kelan loppuun tallennettu tunnustieto on loppunimiö (trailer label). Nimiössä voidaan ilmoittaa tiedoston tunnus tai nimi, viimeisin käsittelypäivämäärä, nauhakelan numero jne. Nimiö voidaan sijoittaa myös tiedoston loppuun. Vakionimiöiden tarkistus on käyttöjärjestelmiin kuuluva automaattinen toiminto.

Magneettinauhalla voidaan siis käyttää sekä alku- että loppunimiöitä, jotka fyysisesti sijaitsevat tiedoston edessä ja jäljessä. Levynimiöt taas voivat sijaita missä tahansa paikassa levyllä (käyttäjän määrityksen mukaan).

Nimiöt luetaan muistiin ohjelman alussa ja lopussa. Tällä varmistetaan se, että tosiaan oikeat tietueet on käsitelty. Nimiöiden avulla voidaan myös todeta tiedoston loppuminen, ja sitäpaitsi loppunimiön sisältämää tietueiden lukumäärää voidaan käyttää tarkistuksiin.

Alustustarkistukset

Lähes kaikissa ohjelmissa ensimmäisillä käskyillä suoritetaan erinäisiä aputoimia varsinaista käsittelyä varten. Näitä aputoimia ovat esimerkiksi kytkimien asetus, rekisterien nollaus jne. Lisäksi voidaan suorittaa tarkistuksia - esim. tarkistaa ovatko kaikki ohjelman tarvitsemat syöttö/tulostuslaitteet toimintavalmiita. Aputoimiin kuuluvat myös tiedostojen nimiöiden tarkistus ja päivitys, ohjelman vakioiden laskenta sekä muun systeemin asianmukaisen toiminnan kannalta tarpeellisen informaation valmistelu. Systeemin ja operaattorin välisen yhteydenpidon hoitaminen saattaa tässä yhteydessä olla erittäin oleellinen seikka. Ohjelmoijan työtä helpottaa huomattavasti se, että monet edellä mainituista tarkistuksista kuuluvat käyttöjärjestelmien tehtäviin.

Jatkokohta ja jatkoaloitus
(Checkpoint/restart)

Jatkokohtaproseduuri on aliohjelma, joka suoritetaan määrätyin aikavälein tai määrätyssä ohjelman kohdassa. Tarkoituksena on tallentaa ohjelman (ja koko systeemin) senhetkinen tila siltä varalta, että myöhemmin mahdollisesti jonkin virheen johdosta keskeytyneen ohjelman toimintaa voidaan jatkaa jatkokohdasta tai tarkistuspisteestä tarvitsematta aloittaa koko ajoa alusta. Jatkokohtatietueet (=ohjelma) tallennetaan magneettinauhalle tai levylle. Jatkokohdan tallentamisen jälkeen ohjelman toiminta jatkuu normaalisti.

Erittäin käyttökelpoinen jatkokohtamenettely on pitkissä tietokoneajoissa, jotka näin voidaan jakaa pieniin osiin. Jokainen osa on erillinen ja riippumaton. Virheen sattuessa voidaan toimintaa jatkaa edellisestä tarkistuspisteestä.

Jatkoaloitusrutiini palauttaa tietokonejärjestelmän samaan tilaan, jossa se oli tarkistuspistettä tallennettaessa. Tämä merkitsee esimerkiksi magneettinauhojen kelausta keskeytyskohtaan jne. Toinen jatkoaloitusrutiinin tehtävä on palauttaa keskusmuisti keskeytyshetken mukaiseen tilaan, so. ladata keskeytetty ohjelma muistiin jatkokohtatiedostosta.

Asianmukaisesti käytettynä jatkokohta/jatkoaloitusmenettely omalta osaltaan edistää koko järjestelmän toimintatehoa. Esimerkiksi sähköhäiriön tai vakavan konevirheen sattuessa voidaan keskeytynyttä työtä jatkaa - se, miten usein tarkistuspisteitä tallennetaan, määrää miten paljon joudutaan ajamaan uudelleen - tarvitsematta aloittaa koko työtä alusta. Monissa tapauksissa tämä merkitsee sitä, että koneaikaa voidaan säästää tuntikaupalla.

Jatkokohta/jatkoaloitusmenettely mahdollistaa luonnollisesti myös sen, että ohjelma voidaan keskeyttää muunkin syyn kuin virheen takia, esimerkiksi sen takia, että joudutaan suorittamaan jokin tärkeämpi työ kuin sillä hetkellä koneessa oleva. Siten mikä tahansa ohjelma voidaan keskeyttää, ajaa sen sijasta jokin toinen ohjelma, ja jatkaa keskeytettyä ohjelmaa kun olosuhteet sen sallivat. Tällaista keskeytysmenettelyä voidaan käyttää myös esimerkiksi työvuoron päättyessä, samoin silloin kun tietokonejärjestelmä on korjauksen tai huollon tarpeessa (so. nimenomaan ennalta-arvaamattomat tapaukset).

KONETARKISTUKSET

Tietokoneen toiminta jakautuu kahteen osaan, työn suoritukseen ja tulosten laadun tarkistukseen. Tietojenkäsittelyssä varsinainen työ sisältää laskutoimituksia, luku- ja kirjoitustoimintoja, lajittelua jne. Tarkistusoperaatioilla varmistetaan työn tarkkuus ja oikeellisuus. Ohjelmoija voi itse suorittaa näitä tarkistuksia tai, kuten nykyään on asianlaita, ne sisältyvät automaattisina toimintoina käyttöjärjestelmiin. Pohjimmiltaan on kysymys kahdesta tarkistuksesta, nimittäin:

1. Syöttö- ja tulostuslaitteilla käsiteltävien tietojen tarkistuksesta ja
2. tietokoneen sisällä käsiteltävien tietojen tarkistuksesta. Tähän liittyy automaattisesti koneen toimintakoodien eli käskyjen tarkistus (ts. onko käsky 'laillinen' vai ei), laskualueiden ylitysten tarkistus, etumerkkien kelpoisuustarkistus jne.

Useimmiten ei ole syytä pysäyttää konetta jonkin virheen takia, vaan virhe voidaan käsitellä välittömästi haarautumalla asianmukaiseen rutiiniin tms. Esimerkiksi magneettinauhan lukuvirhe voidaan käsitellä siten, että nauhaa kelataan takaisin (backspace) virheellisen tietueen alkuun ja yritetään lukea se uudelleen. Jos tietue nyt pystytään lukemaan oikein, jatkuu normaali toiminta. Jos taas virhe on pysyvä, voidaan ohjelman toiminta keskeyttää, tai sitten merkitä virheellinen tietue muistiin ja jatkaa käsittelyä.

Joissakin järjestelmissä virheen havaitseminen alustaa välittömästi erityisen rutiinin, jolloin koko tietokonejärjestelmä siirtyy virheenkorjausrutiinien valvontaan. Tämän jälkeen toimintaoikeus jälleen siirtyy keskeytyneelle ohjelmalle, jossa toiminta jatkuu normaaliin tapaan. Tämä toiminta on täysin automaattinen. Toisissa järjestelmissä taas ohjelmoija ja/tai käyttöjärjestelmä hoitaa saman asian erilaisten testauskäskyjen avulla.

C 20 – 1684 – 2 F

Suomen IBM Offset, 1969